

HYLARSMET: A Hydrologically Consistent Land Surface Model for Soil Moisture and Evapotranspiration Modelling over Southern Africa using Remote Sensing and Meteorological Data

Report to the
Water Research Commission

by

Scott Sinclair and Geoffrey Pegram
Pegram and Associates (Pty) Ltd
DURBAN

WRC Report No. 2024/1/12
ISBN 978-1-4312-0392-5

March 2013

Obtainable from

Water Research Commission
Private Bag X03
GEZINA, 0031

orders@wrc.org.za or download from www.wrc.org.za

DISCLAIMER

This report has been reviewed by the Water Research Commission (WRC) and approved for publication. Approval does not signify that the contents necessarily reflect the views and policies of the WRC nor does mention of trade names or commercial products constitute endorsement or recommendation for use.

Executive Summary

Good, up-to-date estimates of Soil Moisture are valuable in a wide range of disciplines, including numerical weather prediction, agricultural applications and flood forecasting. For Flash Flood Forecasting the current Soil Moisture state of small catchments with a short response time is a good indicator of risk, but is not easily measured. One of the major challenges facing providers of Soil Moisture products is validation. This is mainly due to the limited availability and coverage of in situ observation networks.

Because validation of Soil Moisture is important and ground measurements are sparse and difficult to obtain in real time, several studies have tried alternative techniques of validation, inter alia: correlations between river flows and soil wetness; and assimilation of remotely sensed Soil Moisture estimates into a water balance model. In the fore-runner WRC project [K5/1683], we adapted the TOPKAPI hydrological model to South African conditions and applied it to model the hydrology of the Liebenbergsvlei. To validate the Soil Moisture estimates, in the absence of a suitable set of ground probes, we were forced to look elsewhere and fortuitously teamed up with the Technical University of Vienna, who were interpreting data from satellites to estimate Soil Moisture. The model intercomparisons reported there were very encouraging.

Important outcomes of the WRC funded project on Soil Moisture estimation reported here include:

- an automated modelling system that produces country-wide estimates of Soil Moisture state [and actual evapotranspiration as a by-product] at a 3 h time-step on a 12 km spatial grid over South Africa;
- a proof of concept for operational use by the South African Weather Service (SAWS) in their national Flash Flood Guidance (FFG) system, which is the subject of investigation and refinement under a parallel WRC project K5/2068 run by SAWS;
- informing numerous other fields (other than FFG) such as crop modelling, and drought monitoring where Soil Moisture estimates are beneficial; one of the most important is agriculture.

We performed (and continue to do so) routine calculations on a monthly basis for the Agricultural Research Council's Institute for Soil Climate and Water (ARC-ISCW) Umlindi newsletter, initially by hand, but later by creating a computational tool-chain to automate this process in an easy and repeatable manner. Throughout the time that we have been producing monthly Soil Moisture assessments we have been archiving the source data-sets and building on the work done in previous project work to improve the HYLARSMET modelling procedure. The primary evidence of this work is that all of the historical and most recent results are available on our website and regularly updated.

In the fore-runner project, we were forced to validate our Soil Moisture estimates using data gathered from satellites, because there were not enough ground probes available. The methodologies against which we make comparisons of the PyTOPKAPI Soil Moisture estimates, derived in this WRC K5/2024 project HYLARSMET, come principally from a two-year European Union FP7 project called GLOWASIS [Global Water Scarcity Information Service], in which we have been partners, in our capacity of UKZN researchers during the period Jan 2011 to Mar 2013. Although the GLOWASIS products are aimed at water scarcity, one of the core drivers of the methodologies created to obtain measures of this signal was remote-sensing of Soil Moisture and global hydrological modelling of catchments and thus soil water behaviour.

In this project we extend the usefulness of the TOPKAPI model by adding a Green-Ampt infiltration module and make the model and source code freely available on the internet as PyTOPKAPI. Then, we investigate the sensitivity of the PyTOPKAPI hydrological model to systematic bias in the variables

rainfall and evapotranspiration, as well as the physically based soil properties that describe the model behaviour. The model sensitivity is assessed in terms of relative changes in the Soil Saturation Index (SSI), which is defined as the percentage of soil pore space filled by water. The volumetric Soil Moisture content can be calculated from SSI using location dependent soil properties, if required. The model sensitivity is calculated at the 7200 sites for a 2.5 year simulation period with a time-step of three hours. This large spatial extent gives results for a wide array of climates and land properties. It turned out that overall, the sensitivity of the model is a closely linear function of, and the same order of magnitude as (or less than), the forcing/parameter bias. This indicates that the model is robust to errors in forcing/parameters (in the sense that the magnitude of these errors is not amplified by the PyTOPKAPI model).

The results also show that the best estimates of soil water can be obtained by improving estimates of the storage parameters and rainfall forcing. However, the storage parameters must be obtained from static soil property data-sets and we show that there is value in making improvements to the rainfall forcing (in this case NASA's TRMM 3B42RT) for places where it is biased relative to observed rainfall. This work is particularly relevant for model application in ungauged basins, where the quality of forcing variables and physical parameters cannot be calibrated. As a result we sought, found and implemented up-to-date maps of soil properties over the country.

Research for itself is fun, but one of the attitudes that the WRC fosters is to 'get it out there'. We spent a good deal of our time in this project doing workshops, making presentations, answering queries and publishing. Beside the routine work of updating the model software, which was an ongoing task at this stage of its development, the most important and rewarding activity was the Workshop, and the concomitant response of the informed members of the Hydrological community; especially gratifying was their support, demonstrated by their voting with their feet/cars/flights and participating seriously to the efforts we made. We consider that we have had a good three years of establishing our product as practical and useful, not just as an academic research exercise. There are now several groups of researchers making use of the PyTOPKAPI model and our HYLARSMET outputs, both locally and internationally.

Where to from here? There are several areas of hydrometeorology which would benefit from extending the capabilities of the HYLARSMET and PyTOPKAPI formulations. The first is to extend the time period of analysis from the 4 ½ years we have done so far [August 2008 to February 2013]; we could certainly back-calculate to 2000, or possibly earlier, once we obtain the earlier forcing rainfall and Met data. This extension would be valuable for giving a baseline of records to allow the determination of anomalies and changes. Then there is the need to spatially intensify the model to cover more than 1 sq km in every 160 sq km, in order to provide spatial continuity and detail of Soil Moisture and actual evaporation over every square kilometre of the whole country. This would immediately combine HYLARSMET and PyTOPKAPI into one model which would valuably provide near real time runoff information over the whole of RSA, in detail – this would assist in improving and informing flood forecasting, agriculture (e.g. Dry land wheat farming in the Free State) and water resources analysis and planning. Looking further afield, we need to share this bounty with our SADC neighbours; we have shown in this study that this can be achieved with some confidence, because the African soil maps are available and TRMM near real time rainfall is global. With a new home for the computation and data store in Agricultural Research Council's Institute for Soil Climate and Water, this powerful product we have developed will not be lost, but can endure as long as it is useful – the longer the better!

Acknowledgements

We gratefully acknowledge the reference group (and their willing alternatives) for their valuable support and contributions.

They are:

Mr W Nomquphu	:	WRC (Chairperson)
Mr C Moseki	:	WRC
Prof C Everson	:	UKZN
Prof BC Hewitson	:	UCT
Prof MJ Savage	:	UKZN
Prof R Schulze	:	UKZN
Dr F Engelbrecht	:	CSIR
Dr GC Green	:	Advisor
Dr E van Wyk	:	DWA
Mr E Becker	:	SAWS
Mr T Newby	:	ARC then lately CSIR
Mr A Clulow	:	UKZN (alt Dr C Everson)
Mr C Jack	:	UCT (alt Prof BC Hewitson)
Mr N Kroese	:	SAWS (alt Mr E Becker)

Contents

	<i>Executive Summary</i>	iii
	<i>List of Tables</i>	vi
	<i>List of Figures</i>	vii
	<i>List of Acronyms</i>	xiii
	<i>Acknowledgements</i>	xv
1	Introduction	1
2	Obtain and Start Using Updated Soil and Land Use data	5
3	Introduce the Green-Ampt infiltration layer to PyTOPKAPI	12
4	Improve Rainfall and ET forcing of HYLARSMET	24
5	SSI Estimates of HYLARSMET on SAFFG Catchments	43
6	Back calculate SSI estimates over the Period of Available Data	59
7	Validate SSI estimates against Other Models	71
8	Maintenance of HYLARSMET Community & HYLARSMET Outreach	84
9	Conclusion	107
	References	109
	Appendix	112

List of Tables

Table	Caption	Page
4.1	ΔSS / pooled statistics. Each variable has been in turn increased and decreased by 10% to compare with the control.	26
4.2	Spatial minimum, maximum, means and standard deviations of ΔSS / values, time averaged at each model cell location, displayed in the maps shown in Figures 4.1 to 4.5 following. These numbers are summaries of the data at each site	27

List of Figures

Label	Description	Page
2.1	Image of monthly average of 3-hourly simulated SSI for January 2010, prior to update. The algorithm version used is v0.3 (Final results from Pegram et al., 2010) and the missing data over the Eastern Cape are due to missing information in the SIRI landtype data-set used to derive the parameters: soil depth and saturated Soil Moisture content.	5
2.2	Map of South Africa in the Lambert Azimuthal Equal Area projection (LAEA) chosen to map the TOPKAPI parameters at 1 km ² resolution over South Africa. The grey dotted gridlines show the layout of a regular latitude/longitude grid in this projection. The region enclosed by the red gridlines is the same as that shown in Figure 2.1, while the blue rectangle represents the minimum bounding rectangle on a regular grid defined in the LAEA projection. All parameter maps have been sampled on a regular 1 km grid in the LAEA projection – the area of each grid cell is equal to 1 km ² .	6
2.3	Comparison between the soil depth and porosity (used as saturated Soil Moisture content) derived from the SIRI data-set and the Agrohydrological Atlas (Schulze et al., 2008). In general there is relatively little difference between the porosity derived from each source.	7
2.4	Gridded map of saturated Soil Moisture content (porosity) at 1 km ² resolution. Derived from Schulze et al. (2008).	8
2.5	Gridded map of soil depth at 1 km ² resolution. Derived from Schulze et al. (2008).	8
2.6	The Southern African portion of the SRTM DEM (Jarvis et al., 2008) at 1 km ² resolution.	9
2.7	Gridded map of slopes derived from the SRTM DEM at 1 km ² resolution.	9
2.8	Gridded map of Manning's roughness coefficient for overland flow at 1 km ² resolution. Derived from the USGS Global Landcover Classification (GLCC, 1997), as described in Pegram et al. (2010).	10
2.9	Gridded map of saturated hydraulic conductivity at 1 km ² resolution. Derived from WR2005 (Middleton and Bailey, 2009).	10
2.10	Gridded map of residual Soil Moisture content at 1 km ² resolution. Derived from WR2005 (Middleton and Bailey, 2009).	11
2.11	Monthly average of 3 hourly simulated SSI for January 2010. The new calculation is on the left and Figure 2.1 is repeated on the right for comparison	11
3.1	Comparison of partition of a fixed depth of rainfall in the original TOPKAPI and New PyTOPKAPI models. The column on the left shows the behaviour of the old model and the one on the right, the new model. The upper row shows the effect of short duration high intensity rainfall while the lower row shows the response of the models to a lower intensity rain rate; all total rainfall depths during the interval are the same (because the depths are the same, the intervals must be different, but still represent a single model time-step). The area of each arrow suggests the amount of flux: blue, biscuit, green and mauve colourings respectively signify rainfall, infiltration, surface runoff and drainage.	13
3.2	Map of Bubbling Pressure ψ_b (mm) at 1 km ² resolution in an equal area projection. The values of ψ_b have been estimated by combining the soil texture classes reported in Middleton and Bailey(2009) with the measured data provided by Rawls et al. (1982)	15
3.3	Map of the Pore Size Distribution Index λ at 1 km ² resolution in an equal area projection. The values of λ have been estimated by combining the soil texture classes reported in Middleton and Bailey (2009) with the measured data provided by Rawls et al. (1982)	16
3.4	Schematic showing the water transfers for a single cell in the original PyTOPKAPI model formulation (a) and for the revised model formulation (b). The original formulation employs only saturation excess to provide inflows to the overland store, while the revised formulation achieves this by saturation excess combined with a precipitation excess which is determined by the soil store infiltration, limited by the dynamic infiltration capacity of the soil. The difference in the two models is the FX^2 term shown ringed in (b)	17

3.5	Map of the Mean ΔSSI (SSI differences between the original PyTOPKAPI model formulation and the revised model formulation), which includes a Green-Ampt infiltration module.	19
3.6	Map of the Standard Deviation of ΔSSI (SSI differences between the original PyTOPKAPI model formulation and the revised model formulation), which includes a Green-Ampt infiltration module.	19
3.7	Map of the Coefficient of Variation of ΔSSI (SSI differences between the original PyTOPKAPI model formulation and the revised model formulation), which includes a Green-Ampt infiltration module.	20
3.8	Plots of the Potential Infiltration Rate (mm), with wetting front depth ranging from 100 to 400 mm, for two different soil types. The sequence of blue shaded lines shows the reduction in potential infiltration with increasing depth of the wetting front for a sandy soil (Sa). The potential infiltration is relatively insensitive to effective saturation and is consistently high. The orange shaded lines indicate the potential infiltration for a mixed Sandy Clay Loam and Clay soil (SaClLm-Cl). In this case there is much larger variation with effective saturation and the potential infiltration is very low when effective saturation is high.	21
3.9	Soil and Overland flow response to a constant rainfall input, whose rate doubles in each successive panel starting at 12.5 mm/hr. Upper 4 panels: Original model formulation; Lower 4 panels: Revised model formulation. Note that in the first four panels, overland flow (green squares) does not occur prior to saturation of the soil store, while in the lower 4 panels overland flow occurs prior to saturation of the soil store when the rain-rate is sufficiently high to initiate surface ponding. Flows on a log-scale due to the shallow slope of the cell shown in this example. The red vertical bars show saturation time.	22
4.1	Temporal mean SSI differences computed at each PyTOPKAPI site over the 2.5 year simulation period between the original simulation and a simulation with soil pore space ($\eta_e = \theta_s - \theta_r$) decreased/increased by 10 %. The Orange/Brown coloured map represents a negative ΔSSI and the Blue map a positive ΔSSI in all these figures.	27
4.2	Same caption as Figure 4.1, with soil depth (L) decreased/increased by 10 %.	28
4.3	Same caption as Figure 4.1, with rainfall (R) decreased/increased by 10 %.	28
4.4	Same caption as Figure 4.1, with saturated hydraulic conductivity (K_s) decreased/increased by 10 % – note the switched sign of the response.	28
4.5	Same caption as Figure 4.1, with actual evapotranspiration (ET_a) decreased and increased by 10 %.	29
4.6	The histograms of the values summarised in Table 4.1.	29
4.7	Eta sensitivity. The first three panels show the ET_0 , the current SSI state and the resulting ET_a on the same day in 2008. The fourth panel shows the ΔSSI map as a result of introducing the Green-Ampt infiltration module.	31
4.8	P[0] at each TRMM site (25 km square) from observations during 1/8/08 to 30/6/00	31
4.9	Locations of the 105 DWA rain gauges offered for this study. The red-ringed sites are in the Liebenbergsvlei to be referenced in the discussion on Figure 4.14. The two green-ringed sites (stations A2E022 and A2E026) and the orange-ringed site (A6E005 to their Northeast) are discussed in Section 4.3.2	32
4.10	DWA's Drainage Regions, labelled	33
4.11	Two sets of double mass plots. Two pairs of three gauges from the same region (B7E006 plotted against B8E005 & B8E008), in the upper row (gauge~gauge). In the lower row, the data from the matching TRMM pixels containing the three gauges have been compared in contemporaneous double mass plots (TRMM~TRMM).	33
4.12	Mean diurnal cycle of precipitation amount, frequency and intensity for the 'Vlei for January February and March. [Figure A1.5a. in the MAHYVA project report WRC 1747/1/10; Rouault et al., 2010]	34

4.13	Left: A comparison of running accumulations of rainfall at station C8E008 and the TRMM pixel it lies in, over the 2008-10 period; the site of the gauge is red-ringed in Figure 4.9. TRMM estimates are routinely higher than gauge at this site. Right: A double mass plot of station C8E009 versus C8E008 (both at Sterkfontein Dam, lying within a kilometre of each other), showing excellent 1:1 similarity.	35
4.14	Comparison of FDFs of a SAWS gauge against three DWA gauges averaged, grouped in the red ring in Figure 4.9. The vertical axis is cumulative relative frequency.	35
4.15	Station sites from WRC K5/305: location of 5070 stations with at least 20 years of data. [McNeill et al., 1994.]	37
4.16	Comparison between DWA data for 1-10-2008 to 30-8-2010 against WRC/305 SAWS simulations for 10 years based on data prior to 1992. Top left FDFs of DWA gauges A2E022 and A2E026 (appearing in the green ring in Figure 8) plotted coaxially. Top right, the 10-year simulations from K5/305 for the nearest SAWS gauges to the DWA ones. Both pairs are comfortably similar. Below, left and right, the DWA and SAWS FDFs from above plotted against their neighbours.	38
4.17	Plot of DWA gauge A6E005 (orange ringed in Figure 4.9) versus its nearest neighbour SAWS simulations in the top left panel. Remaining 3 panels: contemporaneous accumulations of rainfall for each of the three DWA gauges against the corresponding TRMM (not SAWS) estimates over the 22-month period.	39
4.18	Transformation of a TRMM time series using pooled DWA data from region G. The data from the TRMM pixel containing gauge G1E006 is compared before and after Q-Q transformation, respectively in the left and right pairs of images.	40
4.19	Mean change of SSI (%) due to TRMM correction using DWA gauges at each PyTOPKAPI site. This is the result of the TRMM transformation using the DWA gauges pooled by region (see Figure 10 for labelling of the drainage regions). Left panel: spatial distribution of cell averages; right panel: histogram of pooled 3-hourly values of change.	40
4.20	These FDFs show the result of the TRMM Q-Q transformations using the pooled data for the regions. Region A contains 6 and region G contains 4 DWA gauges pooled in their respective regions to perform the transforms.	41
5.1	A vector polygon representing the portion of the Liebenbergsvlei catchment that we want to model using PyTOPKAPI. The green background shows WR2005 quaternary catchments, the purple overlay polygon is a concatenation of the sub-set of those quaternaries that make up the Liebenbergsvlei. [The bar at the top of the figure is scaled to 50 km]	46
5.2	A 5 km buffer around the initial boundary provided. The buffer is indicated by the wide grey outline. [The bar at the top of the figure is scaled to 50 km]	46
5.3	Demonstration of the automated sink filling routine for the Liebenbergsvlei DEM. The colour-bar on the right describes the elevation in metres a.s.l and the red dots show the locations of sinks at each stage in the process. The sinks on the border are ignored by the filling routine as they cannot be corrected without data in every cell surrounding the sink. The cropped catchment in the fourth panel is the region defined by the flow direction map, and includes all cells flowing into the outlet cell.	47
5.4	Vector version of the stream network obtained from DEM analysis (blue line). The red dot shows the catchment outlet obtained by calculating the position of the intersection of the stream network and catchment boundary. [The bar at the top of the figure is scaled to 50 km]	47
5.5 & 5.6	Left: The portion of the DEM extracted by computing upslope cells from the catchment outlet. Right: Raster version of the channel network, upstream of the catchment outlet.	48
5.7 & 5.8	Left: Example of model parameters extracted. Surface slopes from the DEM. Right: Example of model parameters extracted. Soil depths from the Atlas	48
5.9	A directed graph representing a simple stream network. The numbered squares are the graph nodes and the lines joining the nodes show direction using a thickened end as an arrow head. The calculated Strahler order is show by each node's colour.	49

5.10	A directed graph representing the stream network extracted from the Liebenbergsvlei. The 1 km squares are the DEM graph nodes with Strahler order shown by their colour.	49
5.11	A close up showing the lower part of figure 5.10. Two first order streams join to form a second order stream, which does not change order when joined by another first order stream.	50
5.12	Overview of the SAFFG catchments in South Africa, Lesotho and Swaziland. The highly divided SAFFG catchments are the blue coloured polygons, shown overlaid on the WR2005 quaternary catchments. They are clustered within the areas of coverage of the Irene, Durban, Port Elizabeth and Cape Town radars. There are also SAFFG catchments in the fold mountain region of the Western Cape – these are larger in size because the only available rainfall input there is satellite-, not radar- based. [The horizontal bar denotes 500 km.]	51
5.13	A more detailed view of the SAFFG catchments within range of the Irene radar, showing the four WR2005 tertiary level catchments we chose for the modelling process. The elevation is shown by the green (lower) through dark brown (higher) colour scale. Their labelling and areas appear in the Table below. [The horizontal bar denotes 100 km.]	52
5.14	A more detailed view of the four extracted catchments coloured in Figure 5.13 and their stream networks, obtained via automated GIS analysis of the DEM, as described in Figures 5.4 and 5.5 and supporting text. [The horizontal bar denotes 80 km.]	53
5.15	A comparison of the stream network (i) automatically extracted by the GIS (dark blue) and (ii) the WR2005 stream network (light blue). This catchment is A21, the westernmost of the 4 subcatchments delineated in Figures 5.13 and 5.14. [The horizontal bar denotes 30 km.]	53
5.16	Detail of the SAFFG catchments (green) that have the majority of their area falling within the boundary of a WR2005 Tertiary catchment (red outline of Western A21), shown in Figure 5.15. [The horizontal bar denotes 30 km.]	54
5.17	SSI modelled by PyTOPKAPI at 1 km ² resolution for the Southern C21 tertiary catchment shown in Figure 5.13. This is a snapshot of one of the 3 hour time-steps during the 2 year simulation carried out for this deliverable. [The horizontal and vertical scales are in metres.]	54
5.18	The result of averaging SSI over each of the (approximately 60) SAFFG catchments, which fall inside the tertiary catchment C21. This map is computed using the fine-scale modelled results shown in Fig 5.17.	55
5.19	As in Figure 5.18, but showing results for the Eastern tertiary catchment B20 (containing approximately 80 SAFFG catchments)	55
5.20	SSI is typically a slowly varying quantity. This figure shows the SSI averaged over each of the SAFFG catchments, which fall within tertiary catchments C21 (Southern parts) and B20 (Eastern parts)	56
5.21	Monthly evolution of SSI averaged over (approximately 140) SAFFG catchments lying within tertiary catchments B20 and C21. Each panel is a snapshot of conditions at 00:00 on the first day of the month during 2010. The sequence starts with 1 st January 2010 at the top left panel and proceeds in month order from left to right to end with 1 st December 2010 in the bottom right panel.	57
6.1	Comparison between the SSI results obtained for two different base parameter sets. The SSI shown here is the monthly average of 3 hourly simulated SSI for January 2010. The new calculation is on the right. The new algorithm version used is v0.4 (using the updated parameter maps). There are no substantial differences between the SSI simulations of algorithm v0.4 and those using algorithm v0.3, except that SSI is now a little lower in the Northern Cape due to the deeper soil estimates. The main difference is that the missing soil properties over the Eastern Cape have been infilled by reanalysis using the data from Schulze et al. (2008). These issues are shown ringed by ellipses in the two images.	59
6.2	Updated data-flow diagram showing sources of the dynamic data (left panel) and static data (upper right panel) to produce the main information streams: Soil Saturation Index and Actual Evapotranspiration (bottom right).	60
6.3	The flow chart indicating how a casual user (upper panel) or a more serious user (lower panel) will gain access to the images and/or the data used to create them, stored on the SAHG servers.	62

6.4	Evolution of SSI at 3 hourly intervals during a single day (2008/11/10). Note very little change from one panel to the next, apart from a slight increase in SSI over the central Free State, shown ringed in the last two panels, following heavy afternoon thunder showers	64
6.5	Snapshots of SSI at the start of the first day of each month during 2011. Note the significant seasonal changes after the rains end in the interior, but at a very slow rate with a half- life of the order of 4 to 6 months. It is clear that the Eastern Interior starts wetting up after the October rains.	65
6.6	Annual variability in SSI for snapshots on 1 st January each year.	66
6.7	Annual variability in SSI for snapshots on 1 st July each year, on average much drier than January in Figure 6.6 above (except in the Western Cape).	66
6.8	Three hourly accumulations of actual evapotranspiration (ET _a) for 2 nd February 2009, in millimetres matching the colour code to the right of the panel. Note the high variability in space and time, with negligible or very low values during the night time.	67
6.9	Daily evolution of 3 hour ET _a accumulations (mm), (like those shown in Figure 6.8, so that the scales are the same in each figure), but for the 3 hour period ending at 12:00 on each of the 8 days.	68
6.10	A 1-year sequence of snapshots of 3 hour ET _a accumulations, at a monthly spacing (3 hour period ending 12:00 on the 3 rd of each month). The strong seasonal change over the year is evident in the areas usually experiencing relatively high rainfall – Winter rainfall in the Western Cape and Summer rainfall in the Eastern part of the country. Unlike ET ₀ , ET _a does not occur if the soil is dry.	69
7.1	Some collected additional (unverified) output from PCR-GLOBWB; (a) average soil moisture on January 1; (b) average soil groundwater storage on January 1; (c) average groundwater recharge (mm/year); (d) average yearly groundwater discharge to surface water (i.e. baseflow: mm/day); (e) average surface water temperature on January 1 (°C); (f) average snow cover on January 1 (m water equivalent); (g) average ice thickness on July 1.	72
7.2	Left panel: Grid centre's for each estimate plotted in Lat/Lon. Right panel: Detailed view of a single 0.5° grid square.	75
7.3	Top left panel: Maximum water storage capacity (m) for each grid cell in the upper soil layer of the PCR-GLOBWB model (used to convert the depth of soil water provided into SSI); Top right panel: The same, but for the lower soil layer of the PCR-GLOBWB model; Middle panel: The same for the single store HYLARSMET model; all three at 0.5° spacing; Bottom panel: The same for the HYLARSMET model at the original 0.125° spacing.	76
7.4	From top left by row to bottom right, all SSI for 2009/2/17 with the 0.5° grid overlaid. (a) HYLARSMET, forced by TRMM 3B42RT. (b) TUWIEN SSM (combined ASCAT/AMSR-E Soil Moisture). (c) NEO profile Soil Moisture DRYMON-SWI. (d) PCR-GLOBWB SSI with ERA interim rainfall forcing – soil layer 1. (e) PCR-GLOBWB SSI with PERSIANN rainfall forcing – soil layer 1. (f) PCR-GLOBWB SSI with TRMM 3B42 rainfall forcing – soil layer 1	77
7.5	From top left by row to bottom right, all SSI for 2009/2/17 with the 0.5° grid overlaid. (a) PCR-GLOBWB SSI (ERA forcing – soil layer 2) (b) PCR-GLOBWB SSI (PERSIANN forcing – soil layer 2) (c) PCR-GLOBWB SSI (TRMM forcing – soil layer 2) (d) TUWIEN SWI noise estimate (combined ASCAT/AMSR-E)	78
7.6	Sample daily time-series comparing HYLARSMET and other models over 2 years. In the top three rows PCR-GLOBWB's upper (layer 1) on left and lower (layer 2) store on right. Row 1 ERA forced, row 2 PERSIANN forced, row 3 TRMM forced. In the last row on left NEO profile SM and on right TUWIEN SSM (obviously unfiltered)	79
7.7	Scatter-plot comparing HYLARSMET and PCR-GLOBWB daily values in six images reflecting the behaviour in the upper six images of Figure 7.6. In the three rows PCR-GLOBWB's upper (layer 1) on left and lower (layer 2) store on right. Row 1 ERA forced, row 2 PERSIANN forced, row 3 TRMM forced.	80

7.8a	Map of R^2 computed for each 0.5° grid cell (HYLARSMET versus other). Grey areas due to no data from one (or both) of the data-sets during the study period. Soil layer 1 on left and 2 on right. Top row: (a) and (b) HYLARSMET versus PCR-GLOBWB ERA. Bottom row: (c) and (d) HYLARSMET versus PCR-GLOBWB PERSIANN rainfall	81
7.8b	Map of R^2 computed for each 0.5° grid cell (HYLARSMET versus other). Top row: (e) and (f) HYLARSMET versus PCR-GLOBWB TRMM; Soil layer 1 on left and 2 on right. Bottom row: (g) and (h) HYLARSMET versus NEO profile on left SM and TUWIEN SSM on right	82
8.1	Overview of the PyTOPKAPI web presence. PyPI increases the visibility of PyTOPKAPI as most Python users will look there when searching for software. Github hosts the home page, source code repository and download area. Google groups hosts the mailing list and mailing list archive.	85
8.2	Screenshot of the PyTOPKAPI listing on the Python Package Index (http://pypi.python.org/pypi/PyTOPKAPI/). The PyPI is a fully searchable online directory of software packages (add-ons) available for the Python language and contains approximately 11500 packages (as at 1 October 2010)	85
8.3	Screenshot of the PyTOPKAPI documentation website: http://sahg.github.com/PyTOPKAPI/ The website will be periodically updated throughout the life of the project.	86
8.4	Screenshot of the PyTOPKAPI software download page (http://github.com/sahg/PyTOPKAPI/downloads/). The current model release is version 0.2.0. The red box highlights the number of downloads within 2 days of the PyTOPKAPI release.	87
8.5	Screenshot of the PyTOPKAPI source code repository hosted on Github (http://github.com/sahg/PyTOPKAPI/). The latest development version of the model can easily be downloaded, or the source code can be browsed online. For users with more sophisticated requirements, a local version of the code repository can be downloaded and easily updated to reflect the latest official developments.	88
8.6	Screenshot of the PyTOPKAPI commit history page on Github. This allows easy browsing of all changes to the source code and each code commit to the source code repository can be viewed in detail (see example in Figure 8.7 following).	89
8.7	An example of an individual code commit for PyTOPKAPI. The changes are shown in Unified Diff format and software patches are typically provided in the same manner.	90
8.8	A screenshot of the PyTOPKAPI issues page (http://github.com/sahg/PyTOPKAPI/issues/). The issues page is essentially a task list, which helps the developers to prioritize and keep track of Bugs and Feature Requests for PyTOPKAPI.	91
8.9	A screenshot of the PyTOPKAPI mailing list front page (http://groups.google.co.za/group/pytopkapi/). The list is hosted by Google groups and is open for anyone to join. The group has only been recently set up and doesn't have any members yet.	92
8.10	Example entries from the PyTOPKAPI changelog. The first entry shows the source code changes that corrected a bug and the second entry indicates the additional meta-data added to the existing codebase in order to accurately describe PyTOPKAPI on the Python Package Index (PyPI).	95
8.11	Typical images included monthly in Umlindi	98
8.12	berg-channel-network-graph berg-channel-network-graph-zoom	100
8.13	Map of R^2 computed for each 0.5° grid cell of HYLARSMET versus PCR-GLOBWB ERA. Grey areas due to no data from one (or both) of the data-sets during the study period, or due to at least one of the data-sets having a value of zero throughout the period.	101
8.14	TRMM swath on one day: 26 August 2012	103
8.15	Global TRMM January Average rainfall	103
8.16	Global TRMM June Average rainfall – note the Northward shift over mid-Africa	103
8.17	SMOS estimate of Global Soil Moisture : 20 to 23 June 2010	104
	Images from workshop problem solving session material designed and offered by Dr Sinclair	105-6

List of Acronyms

AMESD	Africa Monitoring of the Environment for Sustainable Development
AMSR	Advanced Microwave Scanning Radiometer
ARC-ISCW	Agricultural Research Council – Institute for Soil Climate and Water
ASCAT	Advanced Scatterometer
CSAG	Climate Systems Analysis Group – University of Cape Town
DEM	Digital Elevation Model
DWA	Department of Water Affairs
ENVISAT	Environmental Satellite
ERS	European Remote Sensing Satellite
ET _a	Actual Evapotranspiration
ET ₀	Reference Crop Evapotranspiration
EUMETCast	EUMETSAT's primary dissemination mechanism
EUMETSAT	European Organisation for the Exploitation of Meteorological Satellites
FDF	Frequency Distribution Function
GIS	Geographic Information System
GRASS	Geographic Resources Analysis Support System
HYLARSMET	A Hydrologically Consistent Land Surface Model for Soil Moisture and Evapotranspiration Modelling over Southern Africa using Remote Sensing and Meteorological Data
LAEA	Lambert Azimuthal Equal Area
LSM	Land Surface Model
METOP	Series of three polar orbiting meteorological satellites operated by EUMETSAT
PyTOPKAPI	Python library implementation of TOPKAPI
RSA	Republic of South Africa
SAFFG	South African Flash Flood Guidance System
SAHG	Satellite Application and Hydrometeorology Group [UKZN]
SAR	Synthetic Aperture Radar
SAWS	South African Weather Service
SIRI	Soil and Irrigation Research Institute
SRTM	Shuttle Radar Topography Mission
SSI	Soil Saturation Index
SWI	Soil Water Index
TOPKAPI	TOPographic Kinematic APproximation and Integration [hydrological model]
TRMM	Tropical Rainfall Measuring Mission
TUWIEN	Technical University of Vienna
USGS	United States Geological Survey

1. Introduction

Background

For flood forecasting, catchment management and planning, crop modelling and drought monitoring, accurate and timely updated estimates of Soil Moisture (SM) and actual evapotranspiration (ET_a) are valuable, but until recently, have only been available at isolated sites and during relatively short-lived monitoring campaigns. Obtaining good estimates of these variables continuously in detail over large areas has not been feasible until the advent of remote sensing. The WRC research project K5/1683, which was the precursor to this proposal, dealt with these issues and provided a firm foundation to build on. The product was a suite of detailed, spatial, real time estimates of SM and ET_a which were routinely calculated in real time and made available as up-to-date images on the SAHG UKZN web-site, together with the field variables which were used to calculate them. It showed great promise, but tests showed that there was much room for improvement, as the errors in the input data streams hampered the quality of the product; this deficiency which was outside our previous ambit is addressed in this project to make the product useful in practice.

Reason for the research

Stakeholders confirmed that the preliminary product generated in project K5/1683 was unique and of great potential value. To realise this potential, the research team was encouraged by the project reference group to extend the research into a follow-on phase in order to make substantial improvements to the existing prototype. Areas we addressed in this project include:

- *Improvements to input data.* Current 3-hour rainfall spatial estimates obtained via the TRMM satellite are evidently biased. Furthermore, the soil data previously utilized were limited and out-dated. This project has gone some way towards rectifying the deficiency.
- *Comparisons with other sources of information.* There was an urgent imperative to ensure thorough validation in order to build confidence of the product among potential user groups.
- We wanted to forge the link between, on the one hand, the isolated Land Surface Modelling (LSM) sites at 12 km spacing and on the other, with hydrological response of: (i) small catchments modelled at 1 km spacing and (ii) large catchments modelled with 8 to 12 subcatchments.

The main products

The extended research endeavour, which was realised and reported herein, was designed around the practical realisation of three interconnected products: (i) near real time, daily, country-wide SM estimates and maps at a suitable scale (ii) near real time daily country-wide ET_a data and maps at similar scale and (iii) a “portable” working software system to facilitate accessing and use of data products, available to organisations, researchers and practitioners, enabled through workshops and in-house training. The core module which enabled this software system is the TOPKAPI hydrological model we have adapted to RSA conditions, from which we developed a prototype isolated Land Surface Model (LSM). The final product is a freely available open source software package called PyTOPKAPI.

More specifically, and at the risk of repetition but in order to set the stage for this report, the contractual Aims of the HYLARSMET project are listed below.

Aims of this project:

- Using further technical innovation, improve the existing methodology and software implementation for modelling Soil Moisture information/maps over South Africa
- Substantially improve input to the model by suitable conditioning of rainfall on observations and exploit other (improved) data bases on soils
- Make our implementation of the HYLARSMET hydrological model freely available to a wider range of researchers and practitioners using open source licensing
- Prepare guidelines for transferring the methodology to other institutions/organizations disseminated through workshops and training
- Provide Soil Moisture and Evapotranspiration data at appropriate scales to institutions responsible for Flood forecasting, Drought monitoring, Crop modelling and Catchment Management.

The report is mapped out as follows, loosely following the order of the project deliverables. The exception is the pooling of the outreach and user maintenance activities at the end of the report. Here follows a list of the Deliverables and their descriptions, together with comments on how these were achieved.

Deliverables

1. Obtain and start using updated soil and land use data from ARC-ISCW

The intent of this deliverable was threefold: (i) to obtain and install the updated the soils data for RSA in order to lay the groundwork for a sensible and easy to use set of parameter maps in the remainder of the project, (ii) with this set, to start new calculations to infill the missing data in the former Transkei and Ciskei regions of the Eastern Cape (where our previous soil maps contained no information) and (iii) to perform updates of the previously calculated daily estimates of Soil Saturation Indices (SSI) and Reference Crop Evapotranspiration (ET_0) over the region, using the new datasets, and make comparisons.

2. Set up source code hosting, bug tracking, mailing list and website for HYLARSMET to enable collaborators to more easily use and contribute to the development of the model that forms the core of the project's work

The primary goal of this deliverable was to put in place some structure that decreases the likelihood of the PyTOPKAPI model falling into disuse and the code becoming outdated. The investment in the model development by the WRC (Pegram et al., 2010) would be wasted if this were to occur. Our belief is that by making the model as easy as possible to obtain and use, there will be greater uptake, and by releasing the model code under an open source license we enable contributors to improve the model and adapt it to their own needs. [The discussion on this deliverable is pooled with 6 and 9 at the end of the report.]

3. Add surface infiltration layer to TOPKAPI formulation and test efficacy

In the original design of TOPKAPI, Liu and Todini (2002) formulated the model in the following way. Whether the soil store was saturated or not, all the rainfall during the computational interval was

added to the soil store. At the end of the computation interval an inventory was made. If the store was full, the excess was given to overland and channel stores. Then, based on the content of the soil store, the lateral seepage/drainage was sent to the downslope soil store. This strategy had two effects:

1. the soil store was always depleted (by at least the drainage amount) at the end of the computation interval
2. surface runoff only occurred when the soil store was full, independent of rain rate

We rectified this deficiency by adding an infiltration module which we carefully tested before inclusion in the model; a journal paper (Sinclair and Pegram, 2013) was one of the outcomes.

4. Improve rainfall and ET forcing

We assessed the sensitivity of the model of SSI to the range of variables describing PyTOPKAPI; we found that the Rainfall dominates the ET as far as sensitivity goes. It remains to build on these results to produce a sound method for routinely performing quantile-quantile transforms of TRMM to ensure an improved rainfall product with which to feed PyTOPKAPI. To do this, we would need to employ a denser data-base of up-to-date raingauge records. Nevertheless, with the resources available, the study has exposed the fact that TRMM, more than any other variable associated with Soil Moisture, needs correction to add value to hydrometeorological modelling in RSA. As importantly, we have provided the prototype of a methodology for making these corrections in an effective and routine way.

5. Set up hydrologically consistent implementations of HYLARSMET on several SAWS FFG target catchments and compute Soil Moisture at fine scale

The work done to achieve this goal consisted of two components 1) develop tools to (partially) automate the process of setting up a PyTOPKAPI model in new catchments, so that fine scale modelling of many SAFFG catchments becomes feasible, and 2) to test these tools by using them to do detailed modelling on a selection of SAFFG catchments. The success of the adaptation has been clearly demonstrated in this report and the tools developed here are already assisting new users of the PyTOPKAPI model. We were able to move on with the remainder of the work, recognising that we had a solid software platform to work on.

6. Has the same heading as, and is pooled with 9.

7. Back calculate SWI estimates to 1 August 2008, using best parameters and forcing data from years 1 and 2.

The original intention of this deliverable was to regenerate Soil Saturation Index (SSI) simulation results, using the updated static parameter maps and forcing variables from earlier deliverables, in a desktop study as a once-off effort. We exceeded this original intention by first doing the back-calculations on a monthly basis for the ARC-ISCW Umlindi newsletter in an ad-hoc way by hand, and then developing a tool-chain to automate this process in an easy and repeatable manner. Throughout the time that we have been producing monthly Soil Moisture assessments we have been archiving the source data-sets and building on the work done in previous project deliverables to improve the HYLARSMET SSI modelling procedure. The primary evidence of this work is that all of the historical and most recent results are available on our website at http://sahg.ukzn.ac.za/soil_moisture/ and can be downloaded by interested users (based on several queries and interactions we've had, this is already beginning to occur!).

8. Validate SM: Compare modelled Soil Moisture to a suitable subset of readily available remote sensing and in situ Soil Moisture estimates

The methodologies against which we make comparisons of the PyTOPKAPI Soil Moisture estimates, derived in this WRC K5/1964 project HYLARSMET, come principally from a two-year European Union FP7 project called GLOWASIS [Global Water Scarcity Information Service] in which we have been partners in our capacity of UKZN researchers, during the period Jan 2011 to Mar 2013. Although the GLOWASIS products are aimed at water scarcity, one of the core drivers of the methodologies created to obtain measures of this signal was remote-sensing of Soil Moisture and global hydrological modelling of catchments and thus soil water behaviour.

9. and 6., Maintenance of HYLARSMET community (includes bug fixes, extensions to model, website maintenance, user interaction, workshops etc.)

Beside the routine work of updating the model software, which was an ongoing task at this stage of its development, the most important and rewarding activity was the Workshop, and the concomitant response of the informed members of the Hydrological community; especially gratifying was their support, demonstrated by their voting with their feet/cars/flights and participating seriously to the efforts we made. We consider that we have had a good year of establishing our product as practical and useful, not just as an academic research exercise.

2. Obtain and start using updated Soil and Land Use data

Overview

The work described in this chapter was threefold: (i) to obtain and install the updated the soils data for RSA in order to lay the groundwork for a sensible and easy to use set of parameter maps in the remainder of the project, (ii) with this set, to start new calculations to infill the missing data in the former Transkei and Ciskei regions of the Eastern Cape (see Figure 2.1 for the areas previously missing), (iii) and to perform updates of the previously calculated daily estimates of Soil Saturation Indices (SSI) and Reference Crop Evapotranspiration (ET_0) over the region, using the new datasets, and make comparisons.

The region affected by missing SSI simulation results is shown in Figure 2.1 and was due to a lack of landtype information in the SIRI (SIRI, 1987) data-set for the former Transkei and Ciskei. The three main work tasks were to 1) update the soil properties derived from the SIRI data-set to cover the entire country 2) to select a suitable map projection to represent the country-wide TOPKAPI parameter maps at 1 km² resolution and re-project all the spatial data onto this grid and 3) to recompute the SSI and ET_0 over the whole country since August 2008.

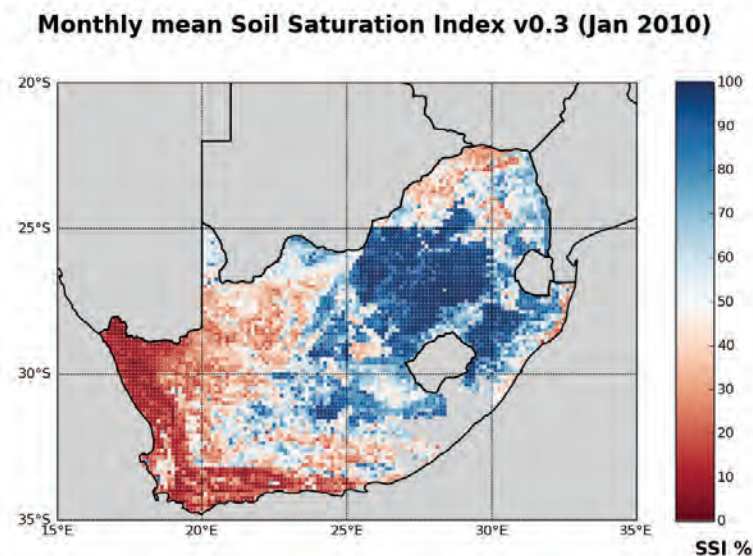


Figure 2.1: Image of monthly average of 3-hourly simulated SSI for January 2010, prior to update. The algorithm version used is v0.3 (Final results from Pegram et al., 2010) and the missing data over the Eastern Cape are due to missing information in the SIRI landtype data-set used to derive the parameters: soil depth and saturated Soil Moisture content.

2.1 Equal area map projection

Mapping the soil and terrain properties in a consistent manner is important. Each TOPKAPI cell is considered to have the same plan area and it was therefore important to choose a map projection which ensures that this property is retained at any location in the modelled region. The family of Equal

Area map projections provide the desired characteristics and we selected the Lambert Azimuthal Equal Area (LAEA) projection to map the TOPKAPI soil and terrain properties over RSA.

All of the input data-sets were projected from their native projection systems to the LAEA projection and the relevant parameter values were then resampled on a 1 km raster grid defined in the LAEA projection. The result is a set of parameter values, which are representative of 1 km² cells, each with equal area on the ground to conform with the assumptions of the TOPKAPI model. Figure 2.2 shows a map of South Africa in the Lambert Azimuthal Equal Area projection (LAEA) projection. This task needed meticulous care and cross-matching of results to ensure success. It is not only crucial for South Africa, but is important for later applications of the Hydrological Model in higher latitudes, where the lat-long grid goes more seriously 'out of square' on the ground.

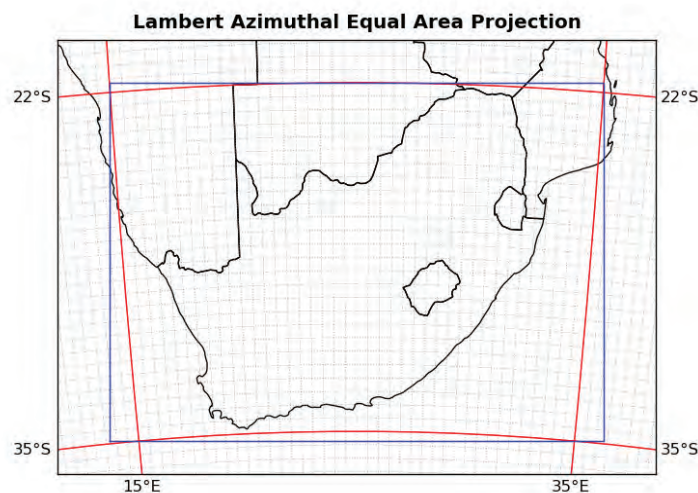


Figure 2.2: Map of South Africa in the Lambert Azimuthal Equal Area projection (LAEA) chosen to map the TOPKAPI parameters at 1 km² resolution over South Africa. The grey dotted gridlines show the layout of a regular latitude/longitude grid in this projection. The region enclosed by the red gridlines is the same as that shown in Figure 2.1, while the blue rectangle represents the minimum bounding rectangle on a regular grid defined in the LAEA projection. All parameter maps have been sampled on a regular 1 km grid in the LAEA projection – the area of each grid cell is equal to 1 km².

2.2 Updating SIRI landtype information

The soil and terrain properties that are used to define and set up the TOPKAPI model for South Africa are derived from a variety of sources (see Pegram et al., 2010 for details). As a result the spatial coverage of each property varies depending on the source data-set used to derive the parameter information. In the initial modelling done for WRC project K5-1683 (Pegram et al., 2010), the soil depth and saturated Soil Moisture content (porosity) were derived from the SIRI data-set (SIRI, 1987). This data-set does not contain any information for the former Transkei and Ciskei and we intended to update it using the revised information contained in the ARC-ISCW Landtype data-set. While working on this, it was realised that the Agrohydrological atlas of Schulze et al. (2008) provided the missing soil properties derived from the ISCW data and so we elected to use this information to infill the missing regions and simultaneously update the soil properties for the whole country. The relevant properties were extracted from Schulze et al. (2008) and resampled onto our chosen grid.

The set of plots in Figure 2.3 provide a comparison between the soil depth and saturated Soil Moisture content obtained from the SIRI data and those derived by Schulze et al. (2008). Both data-sets have

been re-projected to the LAEA projection and sampled onto a 1 km grid as described in the previous section. In general, it turns out that the soil depths from Schulze et al. (2008) are deeper over most of the country, while the differences in porosity are small over most of the country. The places where the porosity differences are large seem to be due to discrepancies in the SIRI data-set, which have been improved in the revised ISCW landtype dataset.

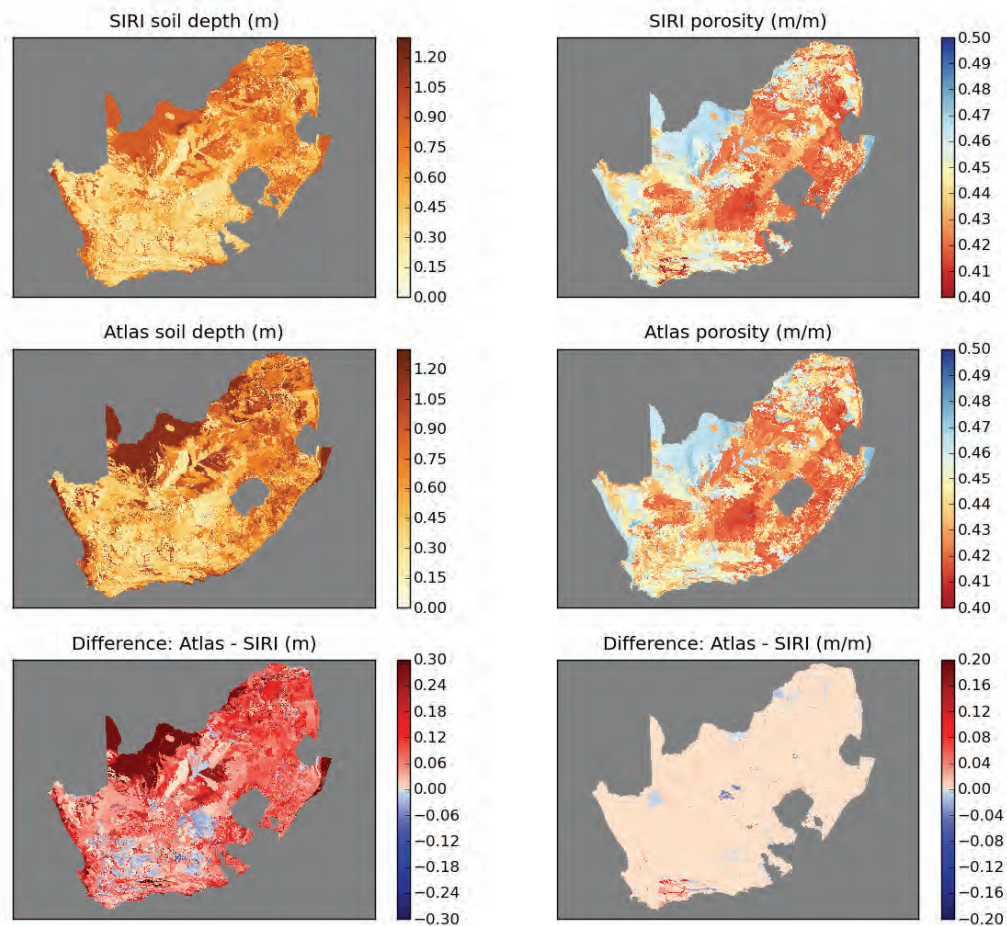


Figure 2.3: Comparison between the soil depth and porosity (used as saturated Soil Moisture content) derived from the SIRI data-set and the Agrohydrological Atlas (Schulze et al., 2008). In general there is relatively little difference between the porosity derived from each source.

The few places in Figure 2.3 which show significant changes are mostly due to anomalies in the SIRI data set. The soil depths obtained from Schulze et al. (2008) are generally deeper over much of South Africa, which results in slightly lower SSI values. This result is due to the way in which the TOPKAPI model works. With a deeper soil of the same porosity, there is a larger pore space; hence a given added depth of rain will give a lower average initial SSI, resulting in a lower ET, which is in turn dependent on SSI. Thus in the long run, the overall effect is that the variability of the SSI is less than expected due to the negative feedback.

2.3 Final parameter maps

In this section, Figures 2.4 to 2.10 show the 1 km gridded TOPKAPI parameter maps in LAEA projection, which were used in the revised SSI simulation algorithm. Finally, Figure 2.11 shows the monthly average SSI for January 2010 obtained by running a simulation using the revised parameter maps, compared with the estimates using the earlier version.

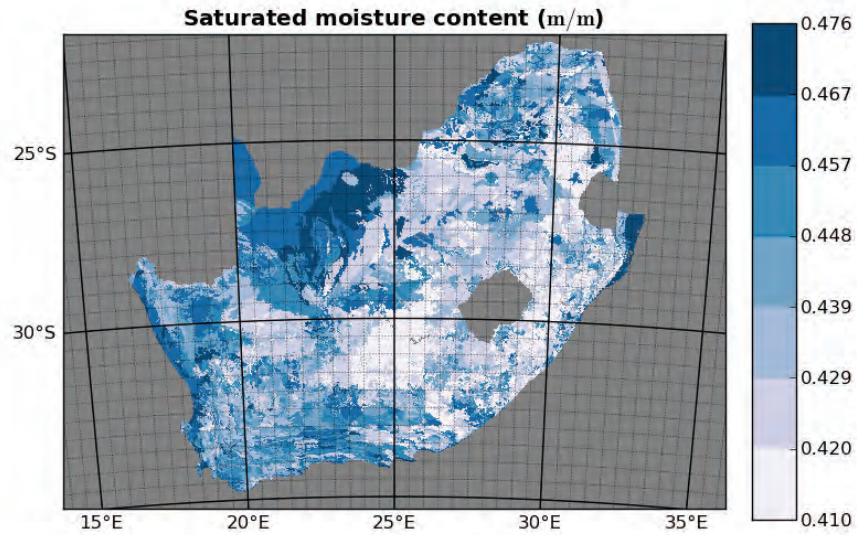


Figure 2.4: Gridded map of saturated Soil Moisture content (porosity) at 1 km² resolution. Derived from Schulze et al. (2008).

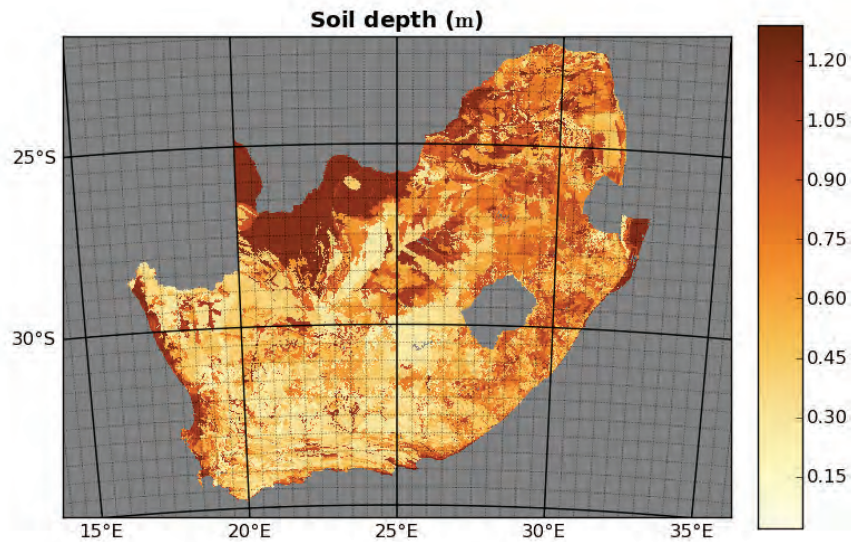


Figure 2.5: Gridded map of soil depth at 1 km² resolution. Derived from Schulze et al. (2008).

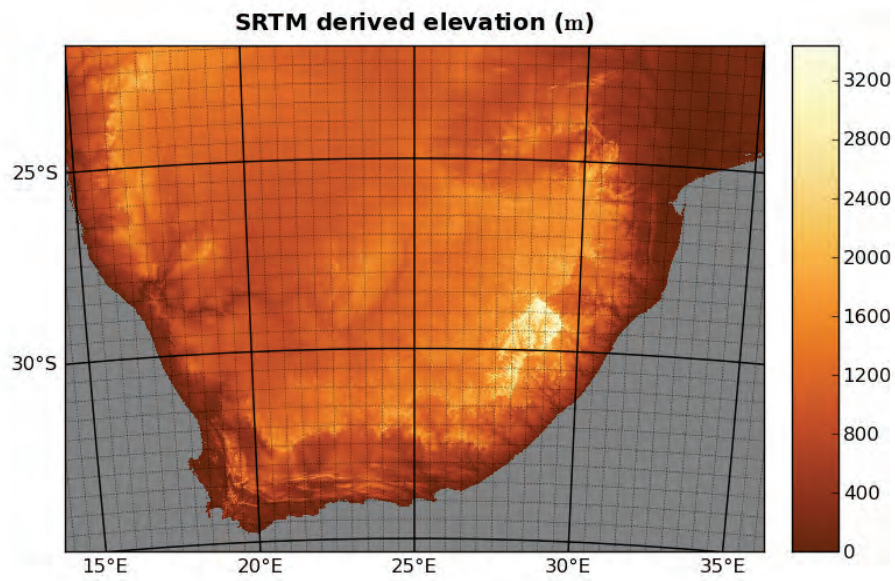


Figure 2.6: The Southern African portion of the SRTM DEM (Jarvis et al., 2008) at 1 km² resolution.

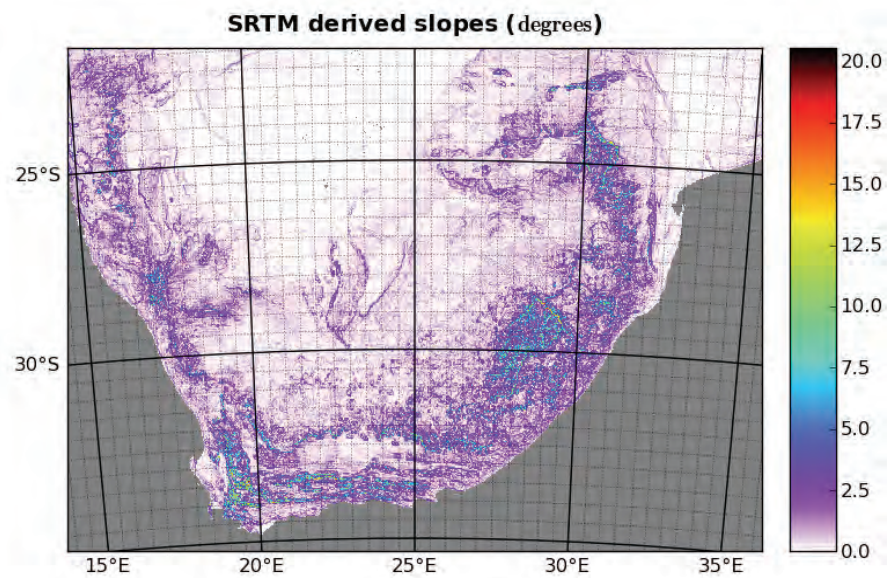


Figure 2.7: Gridded map of slopes derived from the SRTM DEM at 1 km² resolution.

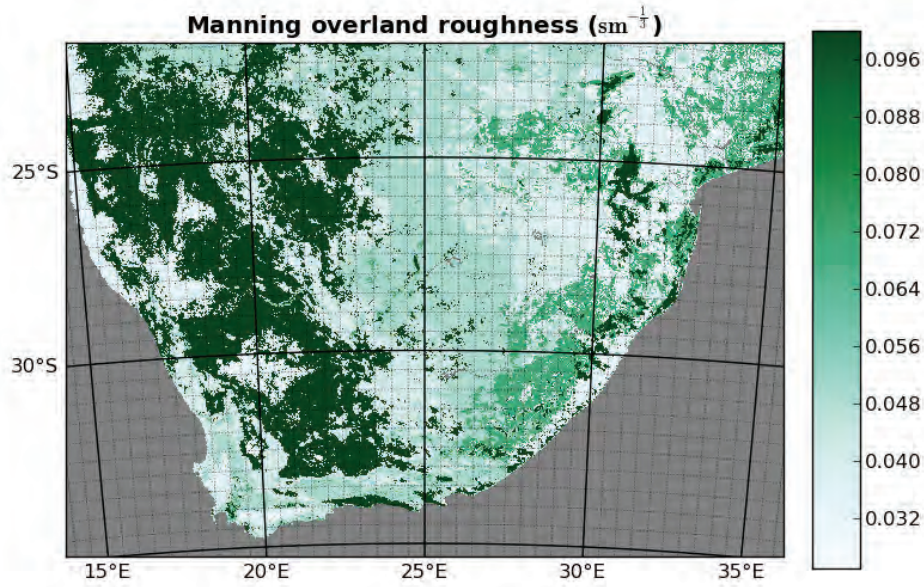


Figure 2.8: Gridded map of Manning's roughness coefficient for overland flow at 1 km² resolution. Derived from the USGS Global Landcover Classification (GLCC, 1997), as described in Pegram et al. (2010).

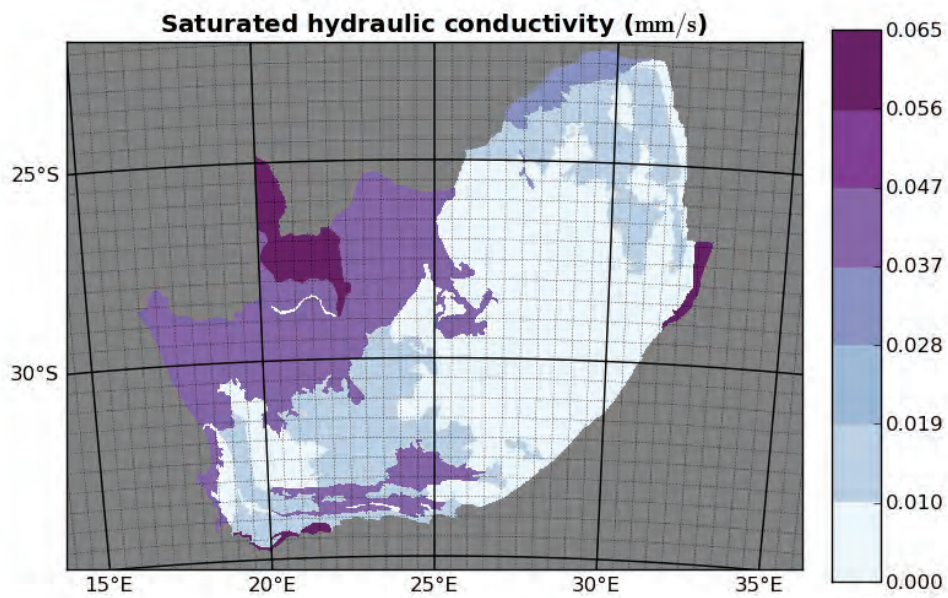


Figure 2.9: Gridded map of saturated hydraulic conductivity at 1 km² resolution. Derived from WR2005 (Middleton and Bailey, 2009).

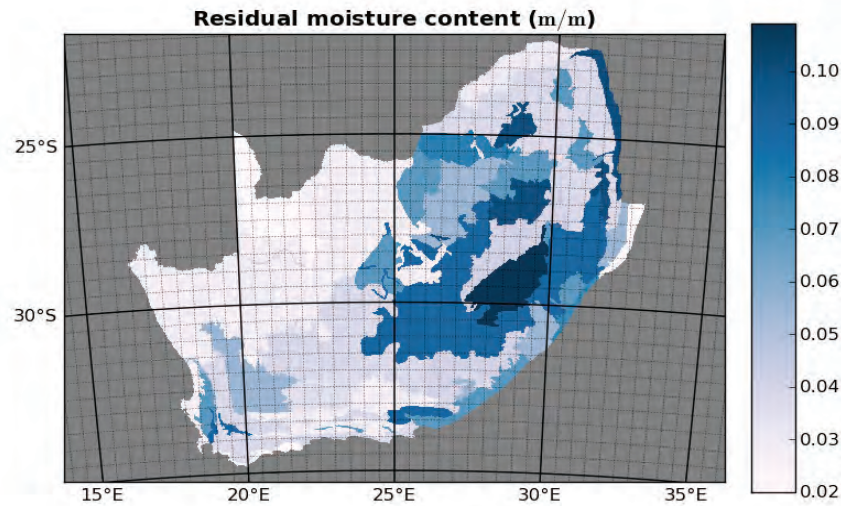


Figure 2.10: Gridded map of residual Soil Moisture content at 1 km² resolution. Derived from WR2005 (Middleton and Bailey, 2009).

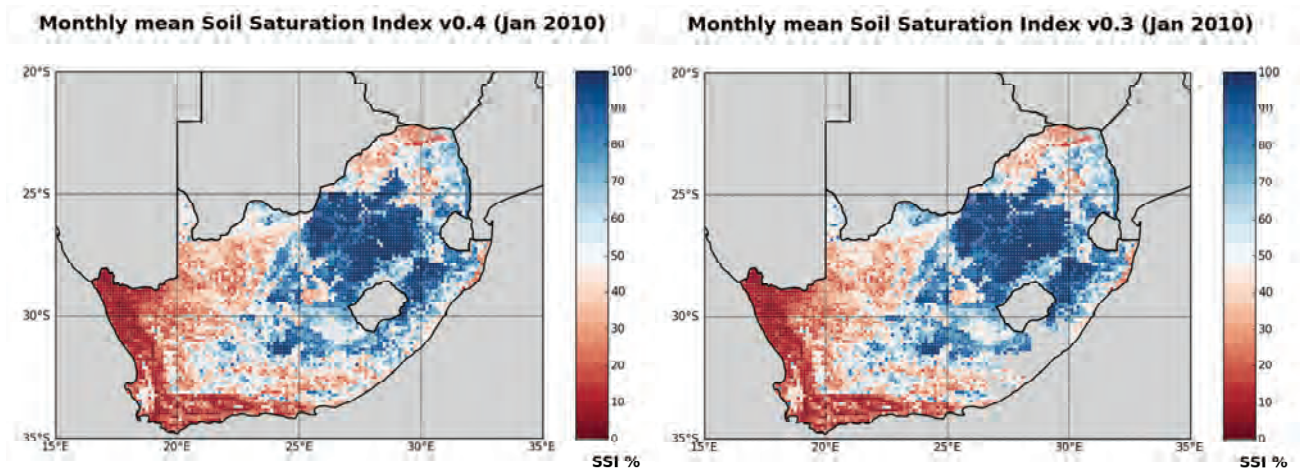


Figure 2.11: Monthly average of 3 hourly simulated SSI for January 2010. The new calculation is on the left and Figure 2.1 is repeated on the right for comparison.

The new algorithm version used is v0.4 (using the updated parameter maps shown in Figures 2.4 to 2.10). There are no substantial differences between the SSI simulations of algorithm v0.4 and those using algorithm v0.3, except that SSI is now a little lower in the Northern Cape due to the deeper soil estimates. The main difference is that the missing soil properties over the Eastern Cape have been infilled by reanalysis using the data from Schulze et al. (2008).

2.4 Summary

The soil and terrain properties used in the TOPKAPI model for modelling Soil Moisture have been updated and now cover the entire country. All of the parameters, previously covering the country and updated, have now been mapped in an equal area projection and gridded at 1 km² resolution to ensure that the TOPKAPI model assumption (of all cells having an equal area) can be satisfied in this and subsequent modelling work. The set of historical daily Soil Saturation Index and Reference Crop Evapotranspiration maps which appear on the SAHG web-site were re-computed and replaced by the new estimates.

3. Introduce the Green-Ampt infiltration layer to PyTOPKAPI

Preamble

This chapter details the work done to introduce the infiltration module into the PyTOPKAPI model.

In the original design of TOPKAPI Liu and Todini (2002) formulated the model in the following way. Whether the soil store was saturated or not, all the rainfall during the computational interval was added to the soil store. At the end of the computation interval an inventory was made. If the store was full, the excess was given to overland and channel stores. Then, based on the content of the soil store, the seepage/drainage was sent to the downslope soil store. This strategy had two effects:

- the soil store was always depleted (by at least the drainage amount) at the end of the computation interval
- surface runoff only occurred when the soil store was full, independent of rain rate

In the earlier adaptation of TOPKAPI to South African conditions (Pegram et al., 2010), the first of these effects was eliminated by modifying the continuity calculation, so that if there was ponding in the overland store, the soil store was left full at the end of the computational interval. The second effect in the model (introduced by an infiltration layer) was what was dealt with in this chapter.

Several methodologies exist for calculating infiltration rates of precipitation into soils. Besides the SCS method, the Horton, Philip and Green-Ampt methods (which are based on the full dynamic Richards equation), are well summarised by Chow et al., (1968). For reasons explained in Section 3.1, the method chosen for application here was the method of Green and Ampt (1911).

The effect this choice has had on the Soil Saturation Index (SSI) behaviour of PyTOPKAPI is summarised in Figure 3.1. There it will be seen that in the original model (left column), irrespective of the rainfall intensity, all rain enters the soil store if there is space (SSI_0) and produces the same residual SSI_1 . In the right column, which illustrates the new adaptation, the infiltration rate depends on: the intensity of the rainfall, and the soil's conductivity and suction head. The lower intensity rainfall is thus going to infiltrate at the soil's capacity to absorb water over a longer interval, so more will go in to the soil than when the same amount of rain falls over a short interval.

The result is that, even on a dry soil, surface runoff can occur if the rainfall intensity is high enough. There is a resulting partition of the rain into infiltration and overland flow. This means that we have successfully adapted TOPKAPI for modelling Flash Flood conditions in small catchments where short time intervals (down to 10 minutes using radar rainfall data) are used in the computation. This has been achieved while maintaining the ability of PyTOPKAPI to model large catchments, which are typically modelled with larger time intervals (3 hours and longer).

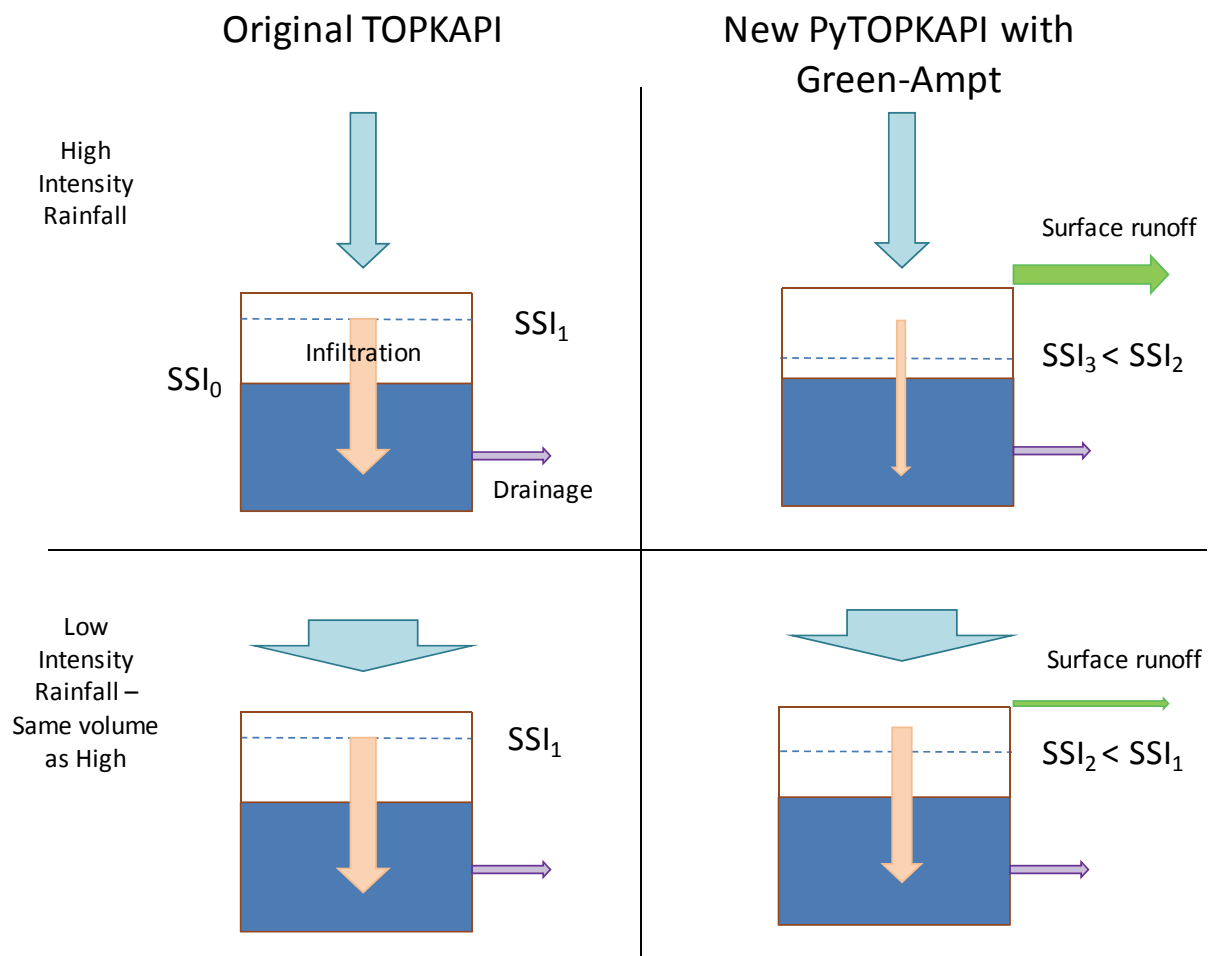


Figure 3.1. Comparison of partitioning of a fixed depth of rainfall in the original TOPKAPI and New PyTOPKAPI models. The column on the left shows the behaviour of the old model and the one on the right, the new model. The upper row shows the effect of short duration high intensity rainfall while the lower row shows the response of the models to a lower intensity rain rate; all total rainfall depths during the interval are the same (because the depths are the same, the intervals must be different, but still represent a single model time-step). The area of each arrow suggests the amount of flux: blue, biscuit, green and mauve colourings respectively signify rainfall, infiltration, surface runoff and drainage.

In Section 3.1, which expands on the development of this methodology and its adaptation outlined here, we will examine the effect this modification has had on our important Soil Saturation Index (SSI) work. It turns out that there is very little difference between the 3-hourly spatial estimates over a large part of the country, as summarised by maps of means, standard deviations (and Cvs) of the daily differences computed over more than 2 years. The noticeable differences are found in places where there is appreciable rainfall, combined with clayey soils which have low porosity (hence low conductivity) – even if they develop high suction heads.

To conclude this preamble – we would like to believe that we have produced a hydrological model for all seasons, sites and applications.

3.1 The Green-Ampt infiltration model

We chose to use the model of Green and Ampt (1911) as the infiltration module for PyTOPKAPI. Apart from the fact that the model is well known and is physically sound, our choice was informed by considering the following factors

- 1) the Green-Ampt parameters could be easily estimated from the information on the soil properties we have available
- 2) it has been shown by Ma et al. (2010) that the Green-Ampt model performs well relative to (i) observed data and (ii) a full 1D solution of Richards equation
- 3) the Green-Ampt model is robust computationally and with a bit of ingenuity, the equations are straightforward to code and include in PyTOPKAPI
- 4) it is 102 years since Green and Ampt's initial publication in 1911; its longevity and continued use in practical applications vouches for its trustworthiness.

In its application to PyTOPKAPI, the infiltration depth during each time interval Δt is calculated using the Green-Ampt equations. Using these, the cumulative infiltration during a precipitation event is determined by solving the equation (Chow et al. 1988):

$$F_{t+\Delta t} - F_t - \psi \Delta \theta \ln\left(\frac{F_{t+\Delta t} + \psi \Delta \theta}{F_t + \psi \Delta \theta}\right) = K \Delta t$$

where $F_{t+\Delta t}$ is the cumulative infiltration depth at the end of the time-step t , F_t is the cumulative infiltration depth at the start of time-step t , ψ is the soil suction head at the wetting front and K is the saturated hydraulic conductivity of the soil. $\Delta \theta$ is the change in moisture content as the wetting front passes and is therefore equal to the difference between the porosity η and the initial Soil Moisture content θ at the start of the infiltration event. If the residual Soil Moisture content is θ_r , then the effective porosity η_e is defined as

$$\eta_e = \eta - \theta_r$$

and

$$s_e = [\theta - \theta_r] / \eta_e$$

so

$$\Delta \theta = (1 - s_e) \eta_e$$

Because PyTOPKAPI works from time-step to time-step by tracking the water volumes in the sets of soil, overland and channel stores, the model keeps track of the effective saturation s_e , which is updated at each time-step; thus the initial accumulated infiltration F_t can be reset to zero at the start of each new time interval. The Green-Ampt cumulative infiltration equation therefore reduces to

$$F_{t+\Delta t} - \psi \Delta \theta \ln\left(\frac{F_{t+\Delta t} + \psi \Delta \theta}{\psi \Delta \theta}\right) - K \Delta t = 0$$

This equation is non-linear in $F_{t+\Delta t}$ and the roots must be obtained by search, or an iterative technique. The equation solver used in PyTOPKAPI is a modified version of the Powell hybrid method Powell (1970), accessed via the Scipy (Jones et al., 2001) wrappers of the MINPACK FORTRAN library (More et al., 1980).

The parameters of the Green-Ampt model are K , $\Delta \theta$ and ψ . K and $\Delta \theta$ are already easily obtainable from the parameters of the original PyTOPKAPI model. However, ψ must be estimated. Since ψ varies as a function of θ and soil type (Chow et al., 1988), it is necessary to obtain a functional form for $\psi(\theta)$ by soil type, in order to describe the time-varying value of ψ with location. El Kadi (1985) evaluated a number of well-known models for $\psi(\theta)$ by fitting them to measured data for a selection of soil samples and comparing the model fits. In general, El Kadi (1985) found that there was relatively little

difference in the model performances, but suggested that the Brooks and Corey (1964) relationship was the model that was most applicable and robust out of those tested. The Brooks-Corey model's applicability in fitting real data is the main reason for selecting it as the model for use in PyTOPKAPI. Its formulation is given by the following equation

$$\theta_e = [\psi_b / \psi]^\lambda$$

or

$$\psi = \psi_b / [\theta_e]^{1/\lambda}$$

where ψ_b is the bubbling pressure and λ is a pore size distribution index for the soil.

A second (more practical) reason to choose the Brooks-Corey model is the availability of model parameter estimates (ψ_b and λ) for a large number of soil samples in the United States produced by Rawls et al. (1982). The Brooks-Corey model parameters are presented by Rawls et al. (1982) for 11 different soil texture classes, which are readily available for South Africa from Middleton and Bailey (2009) and are already used to estimate other parameters in the PyTOPKAPI model (c.f. work reported in chapter 2). The spatial distribution of ψ_b is mapped in Figure 3.2, while Figure 3.3 shows the spatial distribution of λ , respectively obtained by mapping (associating) the geometric means of ψ_b and λ reported in table 2 of Rawls et al. (1982) to the soil texture classes obtained from Middleton and Bailey (2009).

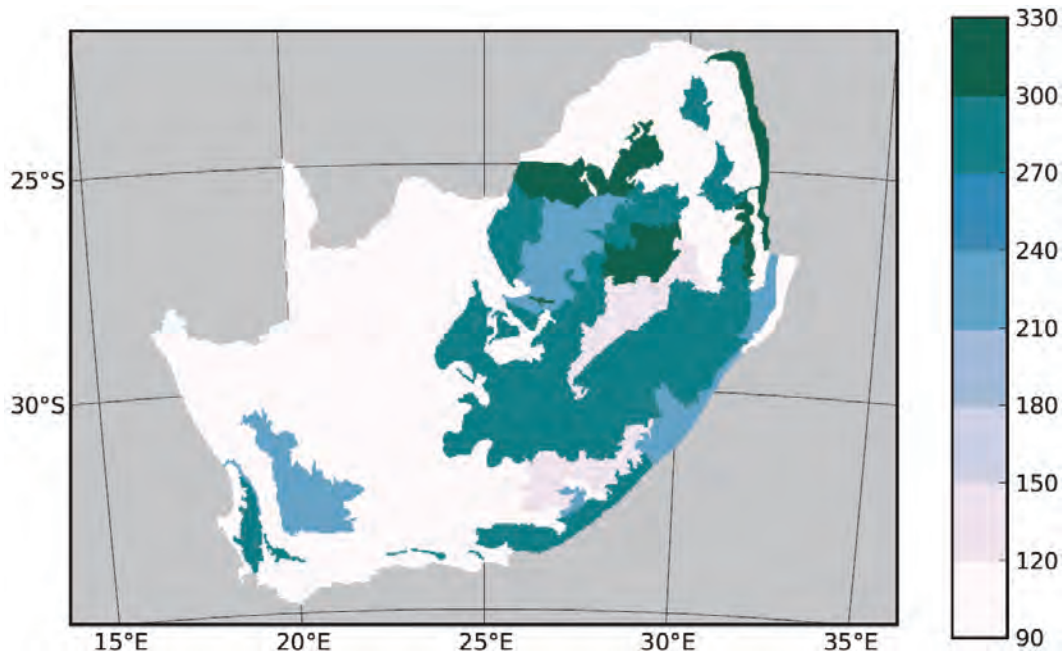


Figure 3.2. Map of Bubbling Pressure ψ_b (mm) at 1 km² resolution in an equal area projection. The values of ψ_b have been estimated by combining the soil texture classes reported in Middleton and Bailey(2009) with the measured data provided by Rawls et al. (1982).

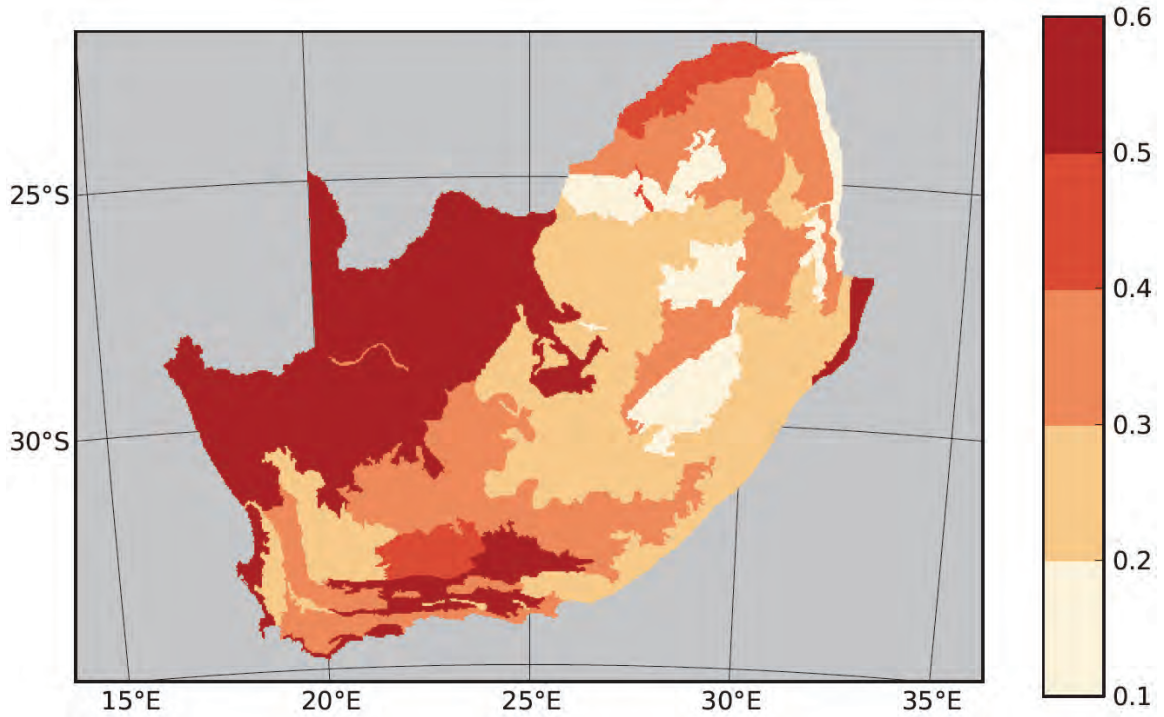


Figure 3.3. Map of the Pore Size Distribution Index λ at 1 km² resolution in an equal area projection. The values of λ have been estimated by combining the soil texture classes reported in Middleton and Bailey (2009) with the measured data provided by Rawls et al. (1982).

3.2 Implementation of Green-Ampt in the PyTOPKAPI model

The first change that was required in the model implementation was a revision of the parameter file format to include ψ_b and λ for each of the country's 1.2 million model cells. This change is not backwards compatible as the new implementation requires the additional ψ_b and λ estimates to run, preventing newer model versions from operating on older parameter files. However, older implementations of the model can still function correctly if they are presented with a “new style” parameter file. Although it is generally felt undesirable to break backwards compatibility, it was unavoidable in this case.

The Green-Ampt algorithm was implemented exactly as outlined in section 3.1 and in addition to coding the algorithm, the logic of water flows within the model had to be significantly modified. These modifications are described in detail below.

In the original model implementation (Vischel et al., 2008a) the total rainfall volume for each time-step enters the soil store directly at a constant rate over the interval. The final volume in the soil store is calculated by solving the differential equation for a generic cell

$$dV/dt = a - bV^\alpha$$

where V is the water volume in the store, a is the constant rate of input (combined from all sources) during the time-step, and b and α are parameters describing the drainage properties of the store [see Vischel et al. (2008a) or Vischel et al. (2008b) for details].

In the original model formulation, all water entering the channel store (if one exists on the cell concerned) comes from upstream channel stores or from contributions by the soil and overland store (see Figure 6 (a)). Water can only exit a channel store via direct evaporation or flow to a down-slope channel store. All other input (precipitation and flow from upstream cells) goes directly to the soil store. The consequence of this is that water can only enter the overland store from the soil store via the mechanism of saturation excess when the soil store becomes saturated during a time-step. In the case of short-duration high-intensity rainfall on wet soils this will not produce the rapid increase in overland volumes expected in practice.

For a given cell in the original formulation, the total inflow rate to the overland store Q_{in}^O is the saturation excess given by continuity as

$$Q_{in}^O = Q_{in}^S - \left(\frac{\Delta V^S}{\Delta t} + Q_{out}^S \right)$$

where Q_{in}^S is the combined inflow rate to the soil store for the current time-step as a result of rainfall, overland flow from up-slope cells and soil drainage from up-slope cells. ΔV^S is the total change in storage for the soil store during the interval and Q_{out}^S is the outflow to the down-slope cell from the overland and soil stores.

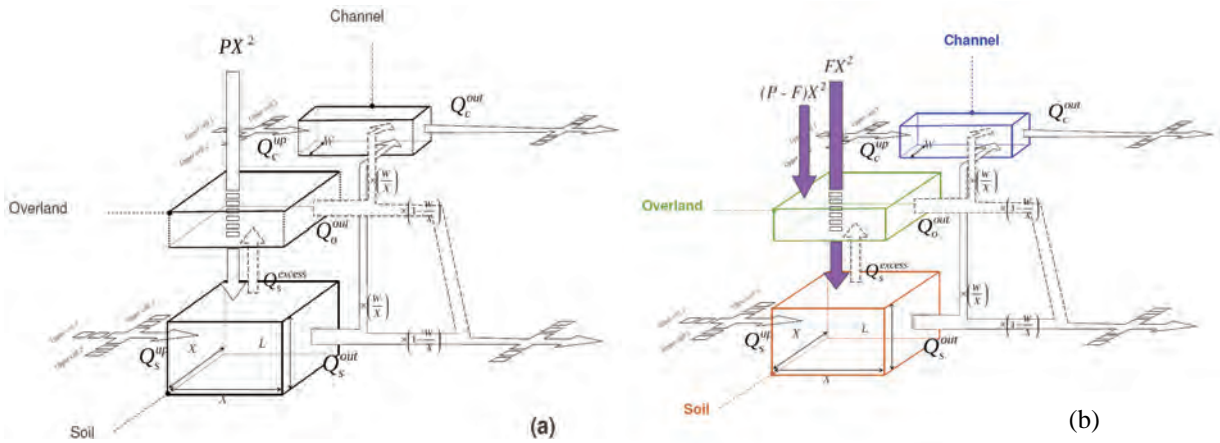


Figure 3.4. Schematic showing the water transfers for a single cell in the original PyTOPKAPI model formulation (a) and for the revised model formulation (b). The original formulation employs only saturation excess to provide inflows to the overland store, while the revised formulation achieves this by saturation excess combined with a precipitation excess which is determined by the soil store infiltration, limited by the dynamic infiltration capacity of the soil. The difference in the two models is the FX^2 term shown ringed in (b).

By contrast, in the newly revised PyTOPKAPI model formulation employing the Green-Ampt infiltration, the model for Q_{in}^S becomes

$$Q_{in}^O = Q_{in}^S - \left(\frac{\Delta V^S}{\Delta t} + Q_{out}^S \right) + P_{excess}$$

where P_{excess} is expressed as a rate, calculated as

$$P_{excess} = P - F$$

P is the average precipitation input rate during the time-step and F is the average infiltration rate into the soil store during the time-step. The P_{excess} term provides PyTOPKAPI with an ability to model the quick overland flow responses from wet soils that cause flash flooding. The new model formulation can be compared with the original by comparing the schematics in figures Figure 3.4 (a) and Figure 3.4 (b). The addition of the Green-Ampt infiltration module affects the initial partitioning of water between the soil and overland stores, particularly for high rainfall rates.

3.3 Comparing the original and revised PyTOPKAPI model formulations

The new model formulation was expected to produce changes from the original model only in circumstances where there were highly intensity rainfall events or where there are soils that have low infiltration potential and are therefore prone to ponding at lower rainfall rates. For low intensity rainfall or soils with a very high infiltration potential (high conductivity soils and soils that are able to develop high suction heads when their effective saturation is low), we did not expect to see much difference between the two model formulations because all of the rainfall would infiltrate directly into the soil store. The models should behave the same where the rainfall is low and the soils have higher conductivity. This needed to be checked on each of the 7000 Soil Saturation Index (SSI) sites.

We chose to test the difference in formulations over the full set of locations and rainfall conditions by comparing multi-year simulations of SSI using each model formulation in turn. The simulation period used in the analysis that follows is from 2008-08-01 to 2011-02-02 with a 3 hour model time-step.

For each model time-step a map of SSI was computed using the revised PyTOPKAPI formulation which was subtracted from the corresponding SSI map produced using the original model formulation. The mean, standard deviation and co-efficient of variation, of the differences in SSI (ΔSSI) at each location, were then determined for each the of 3-hour intervals over the 2 ½ years, which yielded 3 sets of 7446 maps each with 7000 values.

The maps of summary statistics in Figures 3.5 to 3.7 show the results of these calculations. The means of ΔSSI obtained over the simulation period, and shown in Figure 3.5, are relatively small throughout, with values below 0.5% over most of the country and higher values in only a few parts of RSA. We note that the SSI values, which are differenced, cover the full range from 0 to 100%. An estimate of the precision of these estimates of the mean ΔSSI is given by its standard deviation divided by \sqrt{n} , a value with a maximum of $3\%/\sqrt{7446} = 0.04\%$ for all these data. This result of very small differences is to be expected given the 3 hour time-step of the simulations, since the average rainfall rate over such periods is typically low and therefore the majority of the rainfall will infiltrate into the soil store for both model formulations, with some obvious exceptions which need explanation, to be given in the next paragraph but one.

The map of standard deviations of the ΔSSI shown in Figure 3.6 displays a pattern very consistent with that of the means shown in Figure 3.5. The majority of the country exhibits very low standard deviations in the ΔSSI maps, and the areas with higher standard deviations match well with the areas showing higher mean differences but are somewhat more extensive. We also calculated the coefficient of variation (Cv) for the ΔSSI over the simulation period and they appear in Figure 3.7. The Cv is consistently very low (less than 0.5) over most of the country, indicating that the variability of the ΔSSI scales consistently with the mean value of the differences and that the variability relative to the mean is fairly low over the region.

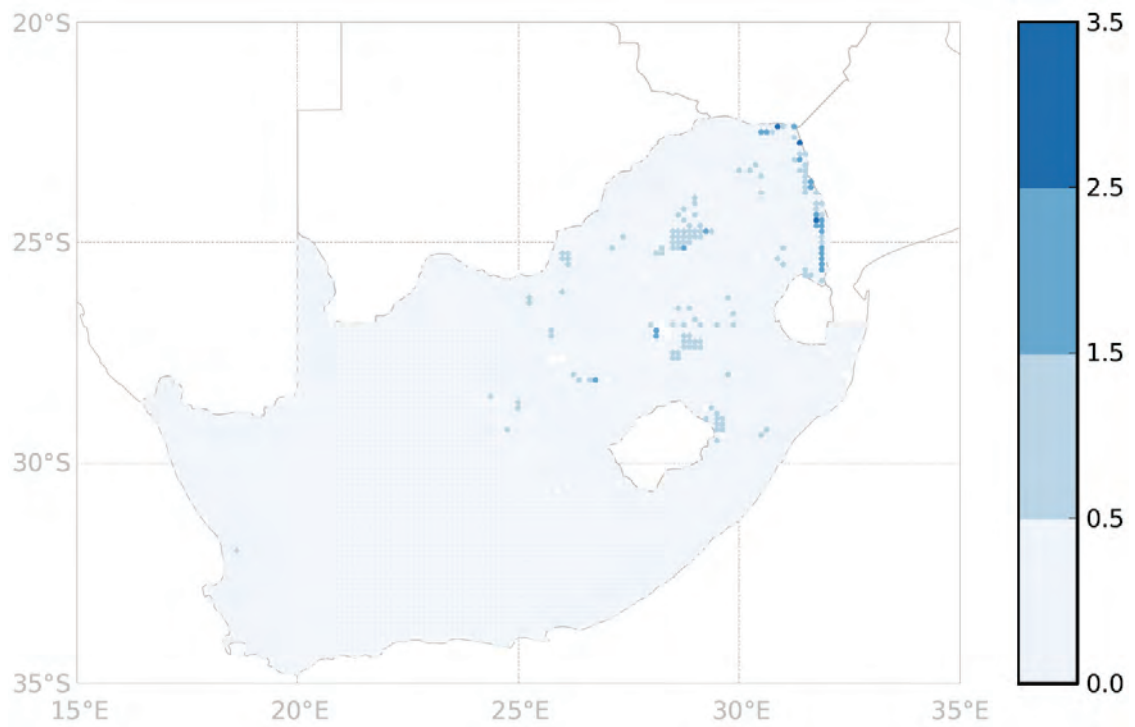


Figure 3.5. Map of the Mean Δ SSI (SSI differences between the original PyTOPKAPI model formulation and the revised model formulation), which includes a Green-Ampt infiltration module.

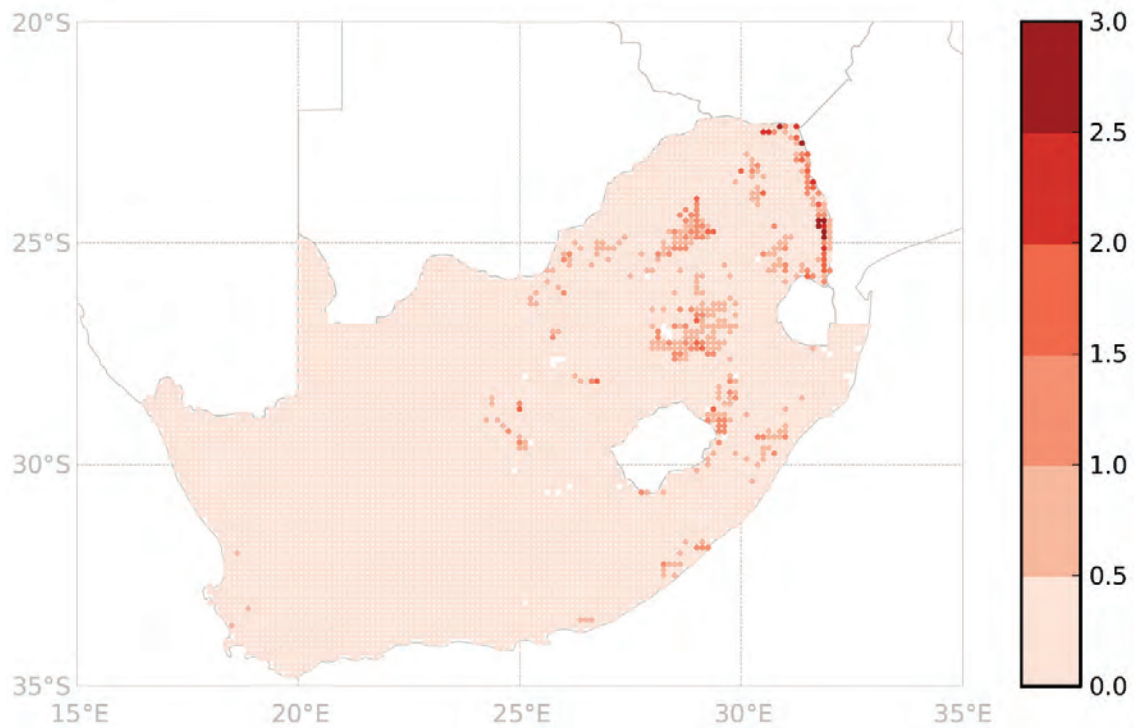


Figure 3.6. Map of the Standard Deviation of Δ SSI (SSI differences between the original PyTOPKAPI model formulation and the revised model formulation), which includes a Green-Ampt infiltration module.

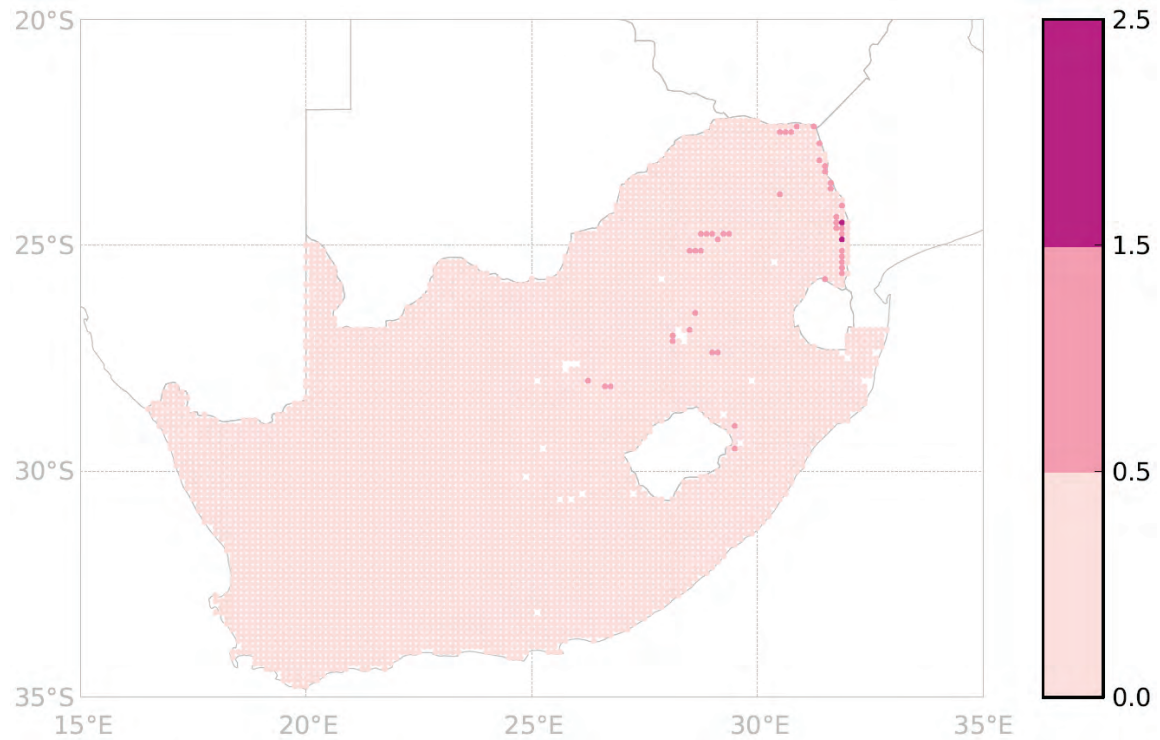


Figure 3.7. Map of the Coefficient of Variation of Δ SSI (SSI differences between the original PyTOPKAPI model formulation and the revised model formulation), which includes a Green-Ampt infiltration module.

The very clear pattern in the map of the standard deviations of Δ SSI is linked to the soil properties in these regions. Note how closely the regions of high bubbling pressure ψ (dark green) and low pore size distribution index λ (pale orange) in Figures 3.2 and 3.3 match those of higher standard deviation in Figure 3.6. The explanation for this lies in the combined effect of the variation of soil suction head ψ with effective saturation s_e and the saturated conductivity K . This combination affects the potential infiltration rate for different values of s_e and explains the patterns seen in Figures 3.5 to 3.7.

3.3.1 Overland flow generation at high rainfall rates

Consider Figure 3.8 showing the potential infiltration rate with wetting front depth for two different soil types. The sequence of blue shaded lines over a range of wetting front depths shows the reduction in potential infiltration with increasing depth of the wetting front for a sandy soil (Sa). The potential infiltration is relatively insensitive to effective saturation and is consistently high, due to the large pore space and high conductivity. In contrast, the orange shaded lines indicate the potential infiltration for a mixed Sandy Clay Loam and Clay soil (SaCCLm-Cl). In this case there is much larger variation with effective saturation and the potential infiltration is very low when effective saturation is high.

The clay rich soils in the regions with darker colours in Figure 3.5 (i.e. which exhibit larger mean Δ SSI values) match the orange line sequence in Figure 3.8. In other words, large changes in potential infiltration rate with effective saturation, can explain the different behaviour of the two model formulations (original and new PyTOPKAPI) in some regions of the country. One consequence is that overland flow is generated at low rainfall rates in these regions, even when we compute at 3 hour time-steps.

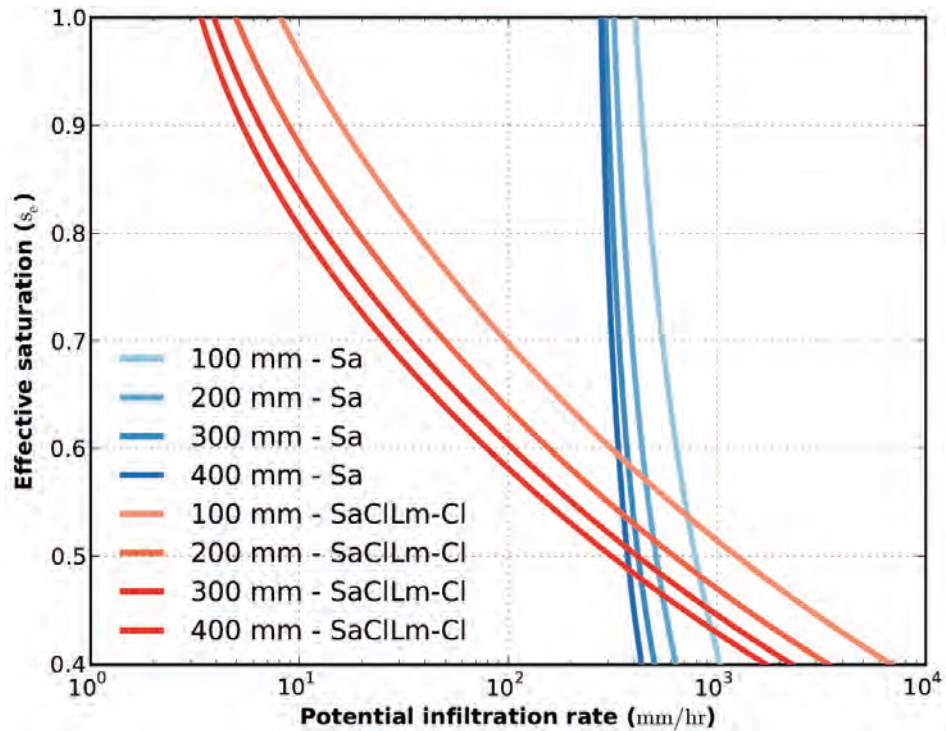
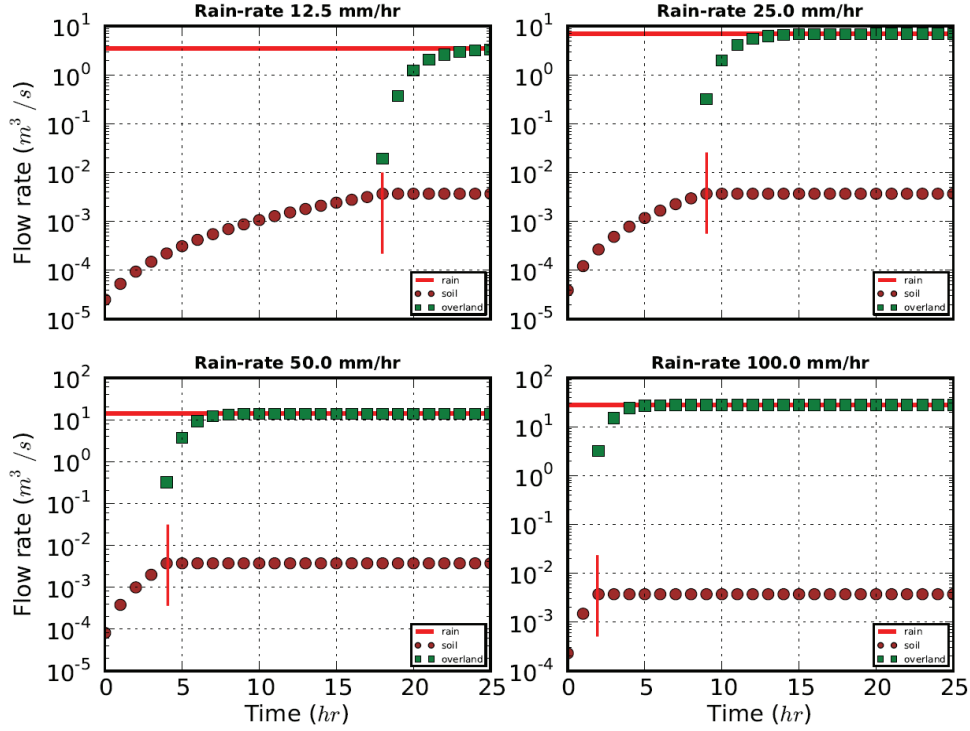


Figure 3.8. Plots of the Potential Infiltration Rate (mm), with wetting front depth ranging from 100 to 400 mm, for two different soil types. The sequence of blue shaded lines shows the reduction in potential infiltration with increasing depth of the wetting front for a sandy soil (Sa). The potential infiltration is relatively insensitive to effective saturation and is consistently high. The orange shaded lines indicate the potential infiltration for a mixed Sandy Clay Loam and Clay soil (SaCILm-Cl). In this case there is much larger variation with effective saturation and the potential infiltration is very low when effective saturation is high.

Figure 3.9 illustrates the mechanism of overland runoff generation prior to saturation of the soil store. A single isolated model cell is subjected to a constant rainfall input, whose rate doubles in each successive panel starting at 12.5 mm/hr. The outflows from the overland and soil stores is computed for

- i) the original model formulation (top four panels), and
- ii) the revised model formulation (bottom four panels).

PyTOPKAPI outflows - Original formulation



PyTOPKAPI outflows - Green-Ampt formulation

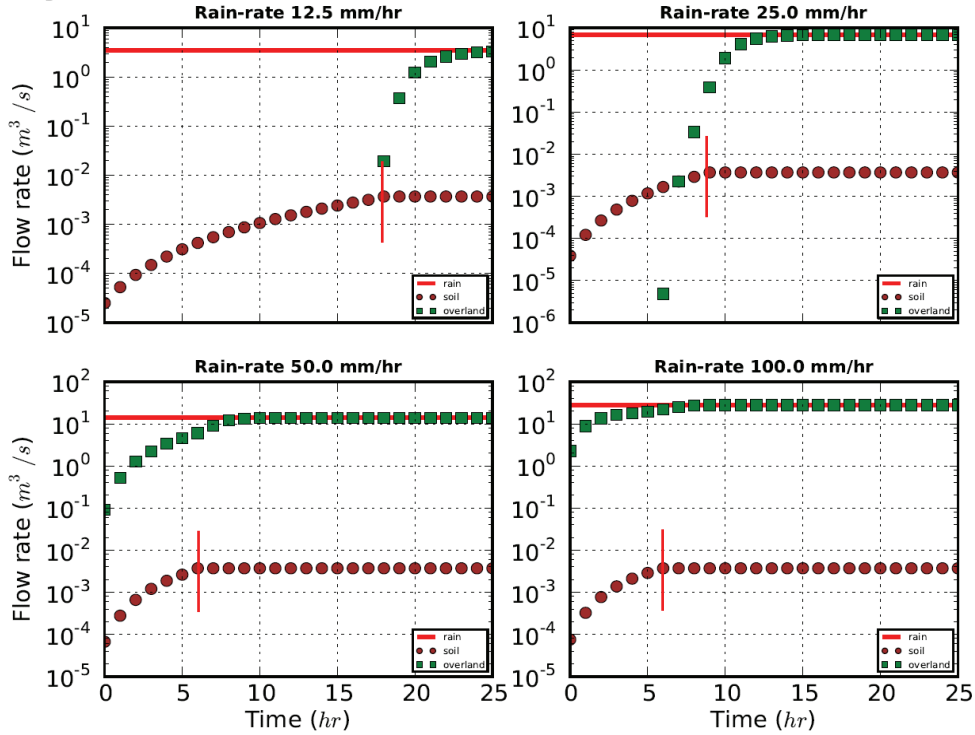


Figure 3.9. Soil and Overland flow response to a constant rainfall input, whose rate doubles in each successive panel starting at 12.5 mm/hr. Upper 4 panels: Original model formulation; Lower 4 panels: Revised model formulation. Note that in the first four panels, overland flow (green squares) does not occur prior to saturation of the soil store, while in the lower 4 panels overland flow occurs prior to saturation of the soil store when the rain-rate is sufficiently high to initiate surface ponding. Flows on a log-scale due to the shallow slope of the cell shown in this example. The red vertical bars show saturation time.

Figure 3.9 shows that overland flow is only generated after full saturation of the soil store, when the original model formulation is applied. This observation is not affected by the rainfall rate. When the revised model formulation is applied, the plots in the lower four panels show that overland flow (green squares) is generated before saturation of the soil store, i.e. when the soil outflow reaches a constant rate as indicated by the small vertical red lines. This phenomenon is observed as soon as the rainfall rate is sufficiently high ("sufficiently high" is determined by the soil type, ground slope and effective saturation at the onset of rainfall). Note that in the panels where the rain-rate is 100 mm/h, the SSI reaches 100% faster in the original formulation than in the new model, because in the latter there is considerably more overland flow before the store saturates. Figure 3.9 shows that direct overland runoff can be generated even at modest rainfall rates with the new Green-Ampt formulation.

In summary, the primary reason for including an infiltration module in the PyTOPKAPI model is to model the rainfall-runoff process in a physically meaningful way. This will allow the model to generate rapid overland runoff when subjected to high intensity rainfall, useful for modelling flash flood situations. The saturation excess methodology of the original model formulation does not generate this feature.

3.4. Summary

A surface infiltration layer based on the infiltration model of Green and Ampt (1911) has been added to the PyTOPKAPI model to introduce a capability in the model to generate rapid overland flows in response to intense rainfall events. This module modifies the original formulation that was solely based on a saturation excess approach to overland flow generation.

The new model formulation has been compared to the original over a long simulation period in land surface modelling mode at a long 3 hourly time-step ("long" being relative to flash flooding time-scales). The two formulations have been found to be very similar in this application, with notable differences only occurring for soils which have low infiltration potential when their effective saturation is high. The significance of this result is that the infiltration module has been shown to generate overland runoff before full saturation of the soil store, as designed. This property of the model makes it very useful for application in flash flood forecasting. In addition PyTOPKAPI retains its usefulness for modelling Soil Moisture state over longer time periods.

What this means is that including an infiltration module in the PyTOPKAPI model allows it to model the rainfall-runoff process in a physically meaningful way. The result is that the model will produce overland flow before saturation of the soil store, when the rain rates are relatively high or when the soils are have low conductivity. True, the original TOPKAPI model was designed to produce overland flow in a way which mimics the variable source area concept, but this only occurs on model cells which were saturated. Notwithstanding this fact, the model has been applied with relative success worldwide (Liu and Todini, 2002). We have made it better by taking its efficacy one step closer to physical reality.

4. Improve Rainfall and ET forcing of HYLARSMET

Preamble

This chapter reports the work done to solve the problem of improving the forcing variables of the PyTOPKAPI model, which have an influence on the Soil Moisture (SM) response of the 7200 isolated sites on a grid across South Africa. The work was done in two stages: (i) assess the sensitivity of SM to variations in the parameters and inputs (ii) improve those data with the greatest influence on the Soil Saturation Index (SSI), the SM surrogate.

Not surprisingly, it turned out that the SSI was most sensitive to the rainfall input and then to the soil parameterization and much less to actual evapotranspiration (ET_a). The soil parameters are fixed (in a sense) by maps of observations which are beyond our capacity to correct. On the other hand, we felt it important to find what steps were necessary to improve (i) the ET_a estimation and (ii) the remote sensing estimates we obtain from NASA's TRMM 3B42RT 3-hour rainfall product.

The report is divided into the following components: (i) Sensitivity of SSI to the following variables: soil parameters (depth, effective porosity and conductivity); rainfall depth; ET_a and (ii) the evaluation and adjustment of the TRMM rainfall input. The last activity was handicapped by the inability to obtain rainfall data from SAWS for the last 3 years. The consequence was that we used 73 raingauge records (selected from the data-base of freely available gauge records on the DWA-Hydrology website) contemporaneous with TRMM, a NASA product, which has had a new version of its rainfall algorithm since 1 August 2008.

4.1. Sensitivity of PyTOPKAPI's Soil Saturation Index (SSI) to the variables defining it

In Sinclair and Pegram (2010) we described how the PyTOPKAPI model has been used to produce estimates of Soil Moisture state at a number of isolated locations across South Africa. The static physical parameters required for the model are obtained from data-sets sampled onto a 1 km² equal area grid covering South Africa. Of the resulting 1.2 million cells, 7200 were selected on a regular latitude and longitude grid with a grid spacing of 0.125°. The selected cells are modelled in isolation, forced by satellite-based rainfall R and an estimate of actual evapotranspiration ET_a as described in Sinclair and Pegram (2010). The near real time rainfall product used is TRMM 3B42RT (Huffman et al., 2007).

The PyTOPKAPI model simulations are run with a 3 h time step and the purpose of the computation is to obtain the Soil Saturation Index (SSI) given in Equation (1). SSI is defined as the percentage of soil void space taken up by water

$$SSI = 100 s_e = 100 (\theta - \theta_r) / (\theta_s - \theta_r) \quad (1)$$

where θ is the Soil Moisture content, θ_s is the saturated moisture content, θ_r is the residual moisture content and s_e is the effective saturation.

Because the physical parameters and forcing variables are estimated by others and are not calibrated by us in this context, it is important to objectively determine the sensitivity of the modelled SSI to

biases in both the parameters and forcing variables in order to focus attention on improving the most influential variable(s).

The relative changes in SSI in response to the changes in parameters and forcing are defined as

$$\Delta SSI = 100[(SSI_p - SSI_c) / SSI_c] \quad (2)$$

where ΔSSI is the relative percentage change in SSI at each modelled location due to SSI_p , which is the SSI calculated by perturbing one of the parameters or forcing variables, and SSI_c is the SSI calculated in the control simulation using default parameter sets.

To obtain these numbers, a control simulation at the 7200 sites in 3 hour intervals over 2.5 years (7300 time steps) was carried out using the default parameters and forcing to obtain the set of SSI_c values. To obtain the set of SSI_p values, a sequence of further simulations was then carried out with each of the parameters and forcing variables increased then decreased by 10 % in turn, with the remainder unchanged; in other words using partial differences. [This was a hugely computer-intensive set of calculations, especially when we were initially trying to find our way to define the problem. A complete run for each change of parameter took approximately 11.2 hours. As there were 15 runs required, (being one control run and two for each of seven variables), and initially each run had to be run at night, a complete analysis took 3 weeks of computing. The suite is now automated and takes a week.]

The physical parameters of the model that can be changed are first the soil parameters: depth L , saturated hydraulic conductivity K_s , saturated Soil Moisture content θ_s , residual Soil Moisture content θ_r , bubbling pressure ψ_b and the pore size distribution parameter λ . The overland and channel Manning roughness parameters of PyTOPKAPI have no effect on the results of this analysis, since in the case of isolated cells there are no channels and overland flow does not have a feed-back relationship with the neighbouring soil store.

The parameters which control storage volume in the soil store are L , θ_s and θ_r . We examined the sensitivity of SSI to changes in L directly, but chose to consider linked changes in θ_s and θ_r by combining them as the effective porosity $\eta_e = \theta_s - \theta_r$. The following relation shows that a proportional change of α in θ_s and θ_r results in an equal change in η_e : $\eta_e = \theta_s - \theta_r$, hence $(1+\delta)\eta_e = (1+\alpha)(\theta_s - \theta_r) = (1+\alpha)\eta_e$, therefore $\delta = \alpha$.

The values of $\theta_s - \theta_r$ were thus increased/decreased by 10% in unison and the results of the combined change reported as changes in η_e . The remainder of the parameters were individually changed.

4.2. The sensitivity analysis

In this section the ΔSSI results are presented in two ways. In Section 4.2.1 the results are pooled for all time-steps and locations, while in Section 4.2.2, the average value of ΔSSI is calculated at each location to yield the spatial variations in sensitivity with soil type, which were then mapped.

The discussion is limited to the forcing variables (R and ET_a), and the physical parameters η_e , L and K_s . The parameters ψ_b and λ required by the PyTOPKAPI infiltration module are only shown in the results summarized in Tables 4.1 and 4.2 because their effect is minimal, as discussed in Chapter 3. This is because the 3 hour time-steps of the SSI simulation usually have rainfall rates that don't exceed the potential infiltration rates for most South African soil types, so that ψ_b and λ don't cause significant changes in ΔSSI .

Summary statistics for all the analyses are given in Tables 4.1 and 4.2. Table 4.1 gives the pooled results for all cells and all locations, while Table 4.2 summarises the results after the ΔSSI values at each cell have been averaged over time.

4.2.1 Pooled results

Table 4.1 gives an ordered summary of the calculated ΔSSI values, pooled over all time-steps and locations. As one might expect, SSI is most sensitive to changes in the parameters that control the available water storage volume in the soil store (effective porosity η_e and soil depth L). A 10% variation in η_e causes an average variation of approximately 9% in SSI, with a standard deviation of around 3% and maximum variations between 30-40%. Variations of 10% in L have an effect of similar magnitude, with an average change in SSI of 8.5%. The next greatest effect is caused by the rainfall forcing R , with an average change in SSI of 4% caused by a 10% change in R . Changes caused by variation in saturated hydraulic conductivity K_s and actual evapotranspiration ET_a are somewhat lower, with changes in these variable of 10% causing average changes of 2.5% in SSI.

Increases in η_e , L and R all cause increases in SSI, and vice-versa. The effects of K_s and ET_a are reversed, with an increase in either of these causing a decrease in SSI. In Table 4.1 each cell's hourly response of SSI (%) was compared with the control value. Each of these differences was recorded for each site. The numbers in the table are the results in percentages pooled over all time-steps and cell locations.

Table 4.1. ΔSSI pooled statistics. Each variable has been in turn increased and decreased by 10% to compare with the control.

	Min	Max	Mean	Std-dev
$\eta_e \downarrow$	-33.36	-0.78	-9.06	3.15
$\eta_e \uparrow$	0.48	42.95	9.37	3.24
$L \downarrow$	-32.11	0.68	-8.48	2.18
$L \uparrow$	-0.16	40.12	8.48	2.28
$R \downarrow$	-10.53	1.29	-4.26	1.89
$R \uparrow$	-1.57	11.07	3.92	1.88
$K_s \downarrow$	-1.43	7.04	2.72	1.58
$K_s \uparrow$	-5.98	1.95	-2.29	1.38
$ET_a \downarrow$	-2.28	35.36	2.50	2.07
$ET_a \uparrow$	-25.89	3.12	-2.44	1.91
$\lambda \downarrow$	-0.23	3.51	0.007	0.06
$\lambda \uparrow$	-3.21	0.52	-0.007	0.05
$\psi_b \downarrow$	-1.28	0.41	-0.008	0.03
$\psi_b \uparrow$	-0.28	1.18	0.006	0.04

4.2.2 Spatial distribution of ΔSSI

To determine the variability in SSI sensitivity to location (due to differences in soil types, slopes etc.), we computed the average value of ΔSSI at each location and summarize the spatial mean, standard deviation, minimum and maximum of these average % values in Table 4.2. The trends and magnitudes of the Mean ΔSSI are very similar to those described in section 3.1 on pooled results, because the quantities in Table 4.2 are a manipulation of the same data that produced Table 4.1.

Table 4.2. Spatial minimum, maximum, means and standard deviations of ΔSSI values, time averaged at each model cell location, displayed in the maps shown in Figures 4.1 to 4.5 following. These numbers are summaries of the data at each site.

	Min	Max	Mean	Std-dev
$\eta_e \downarrow$	-16.07	-6.92	-11.33	0.63
$\eta_e \uparrow$	6.30	17.84	11.47	0.73
$L \downarrow$	-15.21	-5.35	-8.66	1.44
$L \uparrow$	4.95	16.53	8.46	1.53
$R \downarrow$	-9.53	-0.86	-4.49	1.03
$R \uparrow$	0.64	9.36	4.03	1.02
$K_s \downarrow$	-0.36	5.21	3.01	1.36
$K_s \uparrow$	-4.49	0.3	-2.68	1.16
$ET_a \downarrow$	0	11.51	2.81	1.55
$ET_a \uparrow$	-9.48	0	-2.74	1.45
$\lambda \downarrow$	-0.01	0.76	0.007	0.04
$\lambda \uparrow$	-0.72	0.0001	-0.007	0.03
$\psi_b \downarrow$	-0.45	0.003	-0.008	0.03
$\psi_b \uparrow$	-0.004	0.41	0.006	0.03

The set of maps in Figures 4.1 to 4.5 plots the mean of ΔSSI averaged over all 3-hour intervals at each cell location. The spatial patterns for changes in SSI due to variations in L and K_s (Figures 4.2 and 4.4) are strikingly similar (note the difference in scale of ΔSSI). This is in spite of the reversed sign of the ΔSSI due to a positive/negative change in L or K_s ; this similarity is obviously a function of soil type. The relatively smooth behaviour of ΔSSI in Figure 4.1, due to changes in η_e , are confirmed by the separation of the positive and negative values shown in the first histogram of Figure 4.6.

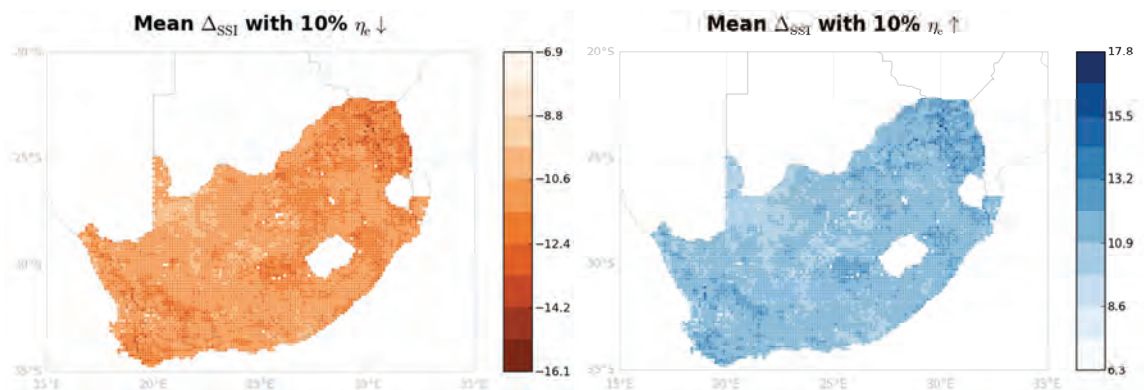


Figure 4.1. Temporal mean SSI differences computed at each PyTOPKAPI site over the 2.5 year simulation period between the original simulation and a simulation with soil pore space ($\eta_e = \theta_s - \theta_r$) decreased/increased by 10 %. The Orange/Brown coloured map represents a negative ΔSSI and the Blue map a positive ΔSSI in all these figures.

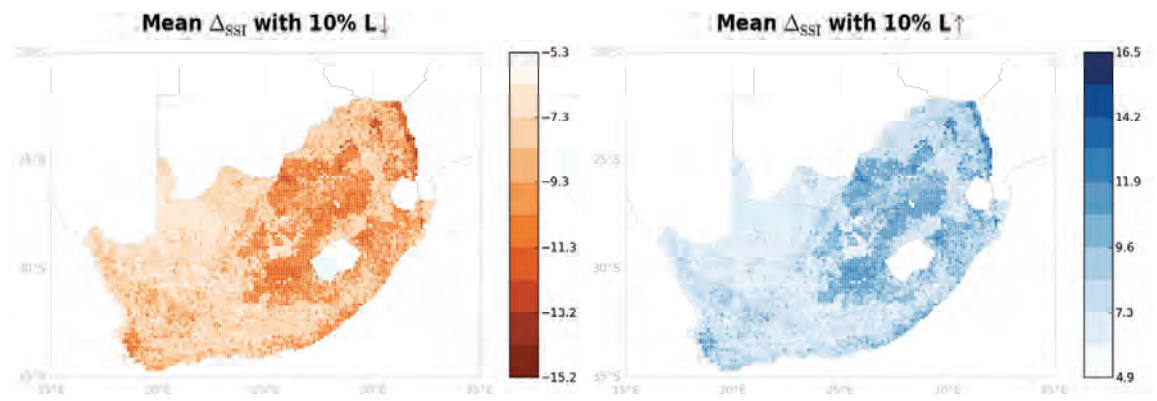


Figure 4.2. Same caption as Figure 4.1, with soil depth (L) decreased/increased by 10 %.

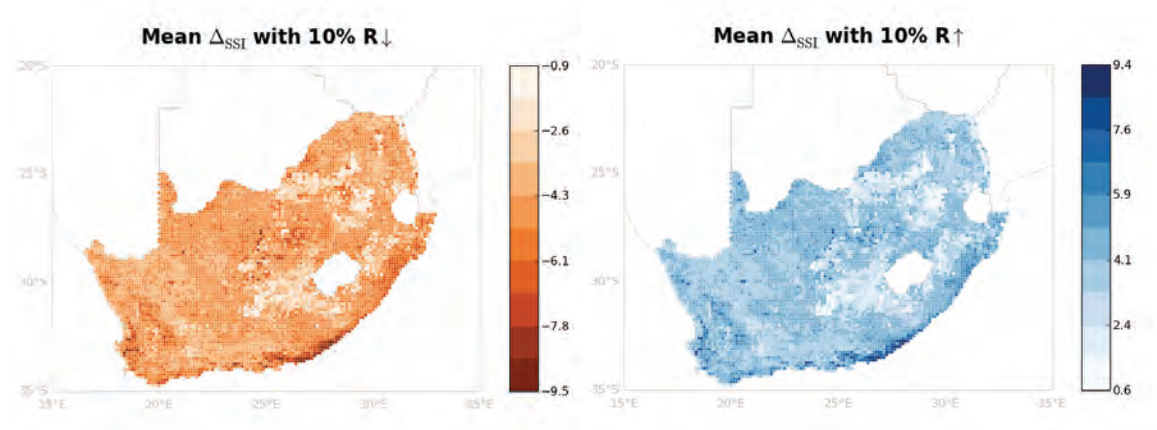


Figure 4.3. Same caption as Figure 4.1, with rainfall (R) decreased/increased by 10 %.

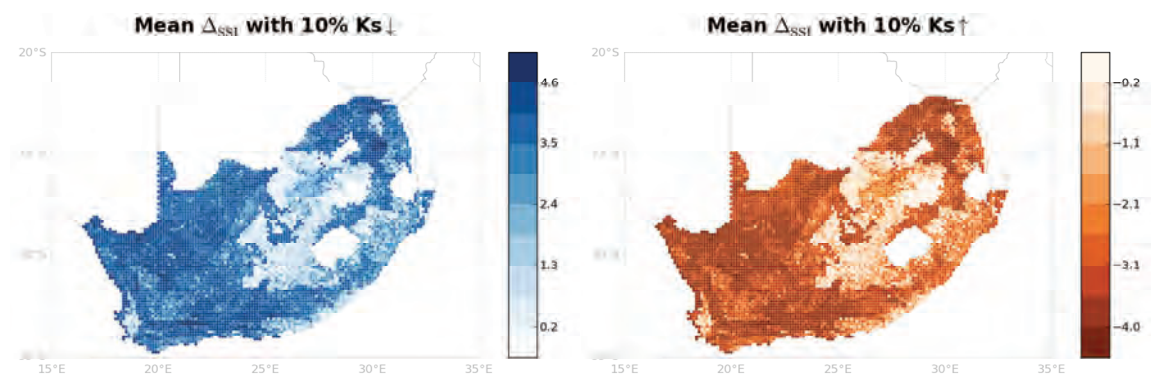


Figure 4.4. Same caption as Figure 4.1, with saturated hydraulic conductivity (K_s) decreased/increased by 10 % – note the switched sign of the response.

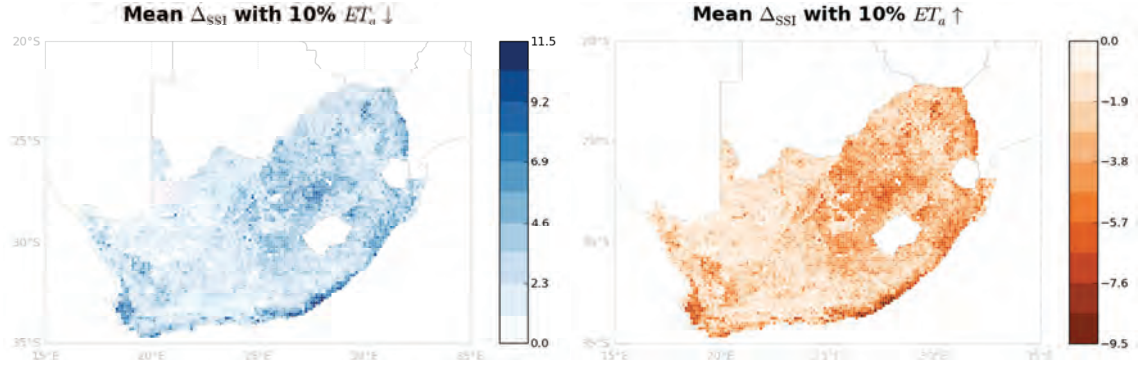


Figure 4.5. Same caption as Figure 4.1, with actual evapotranspiration (ET_a) decreased and increased by 10 %.

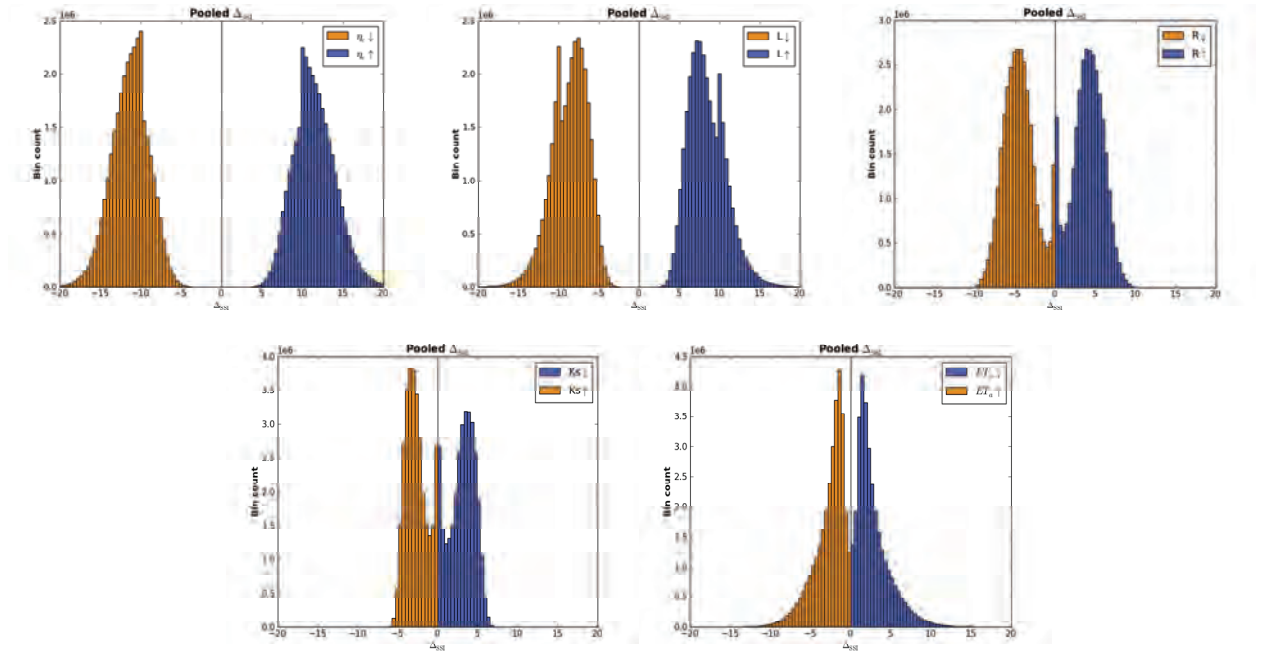


Figure 4.6. The histograms of the values summarised in Table 4.1.

In Figure 4.6, each pair of images show the histograms of the values which were used to compose the summary in Table 4.1; note that the horizontal scales are all the same. The data in Figures 4.1 to 4.5 are summarised in Table 4.2, where the summary statistics reflect near symmetry. This near symmetry of all panels in Figure 4.6 implies an almost linear response of SSI to changes in the variables over the $\pm 10\%$ range.

4.2.3 Summary of results of sensitivity analysis

The response of the soil saturation index to the parameters and forcings by-and-large followed our intuition and is a valuable exercise for deciding where to put one's energy in making the model better. The main response is due to the soil storage parameters, pore space η_e , and soil depth L . These

responses (near 11% and 8.5%) are almost to the same scale (in average) as the change of the parameters, with rainfall R scoring about half that amount; the rest have less effect. How does this information help us?

Soil parameters are collected at isolated sites and then mapped by pedologists; we hydrometeorological modellers trust their expertise and experience. The soil maps are relatively static (there have been minor adjustments over the last few years), so we are bound to accept them. Is uncertainty of soil the parameters bad for hydrology? We think not, because the important point about soil maps is their relative permanence. Even though our hydrological model PyTOPKAPI works in semi-distributed mode, some minor calibration will usually be necessary. Such adjustment to hydrological models to provide satisfactory flow results is already standard practice.

Actual Evapotranspiration, ET_o , is the next issue. From the results of the sensitivity analysis, its response is relatively modest at about 2.8%, when compared to the other variables. The way we performed the sensitivity analysis in this case was to calculate the control ET_o at each 3 hour time step on each PyTOPKAPI site and then factor this amount by $\pm 10\%$, however there is a feedback effect here, because ET_o is very dependent on SSI. This dependence is shown in Figure 4.7, where the first three panels in turn show the ET_o , the current SSI state and the resulting ET_o on the same day. It is clear that the ET_o is extremely dependent on SSI, which in turn is principally affected by rainfall.

In Chapter 3 we introduced the infiltration layer to ensure proper ponding behaviour. When ponding occurs due to heavy rainfall on a soil with low infiltration capacity, in turn dependent on K , θ and ψ , the water evaporates directly from the surface. To show the effect of change in SSI due to the Green-Ampt infiltration module introduced in Chapter 3, we repeat here, in the fourth panel of Figure 4.7, the map of ΔSSI . The sites where evident change has occurred are evidently in the regions experiencing frequent convective rain and where the soil has low infiltration properties. The range of the change is about 3%, the same order as the mean sensitivity overall due to a 10% change in ET_o , given in Table 4.2. Having assessed the sensitivity (and improved the module) of the ET_o determination, we came to the following interim conclusion. The ET_o has a considerably lower effect on SSI than does rainfall (on which it so heavily depends). As a result, we decided to focus the remainder of our analysis on the rainfall estimation via TRMM.

Rainfall is another matter entirely. The SSI response to 10% changes in rainfall, averaged over time, resulted in mean changes of magnitude from 4% to 4.5% with a range of 10 to 11% (see Tables 4.1 and 4.2 and Figures 4.3 and 4.6). On the face of it, this might indicate we need to put more energy into the soil parameters than into the rainfall estimates (TRMM in this case which we will show in Section 4 is biased). However, Figure 4.8 gives the proportion of dry time $P[0]$ in 3 hour rainfall over the country as estimated by TRMM. We note that TRMM produces $P[0]$ values higher than those determined from the daily accumulated values appearing later in the report, because of TRMM's higher sampling frequency and also because gauges accumulate rainfall. It was a surprise that the $P[0]$ in the Highveld interior (600 mm/y) is less than that in the coastal zones and especially the South-western Cape and North-eastern Mpumalanga.

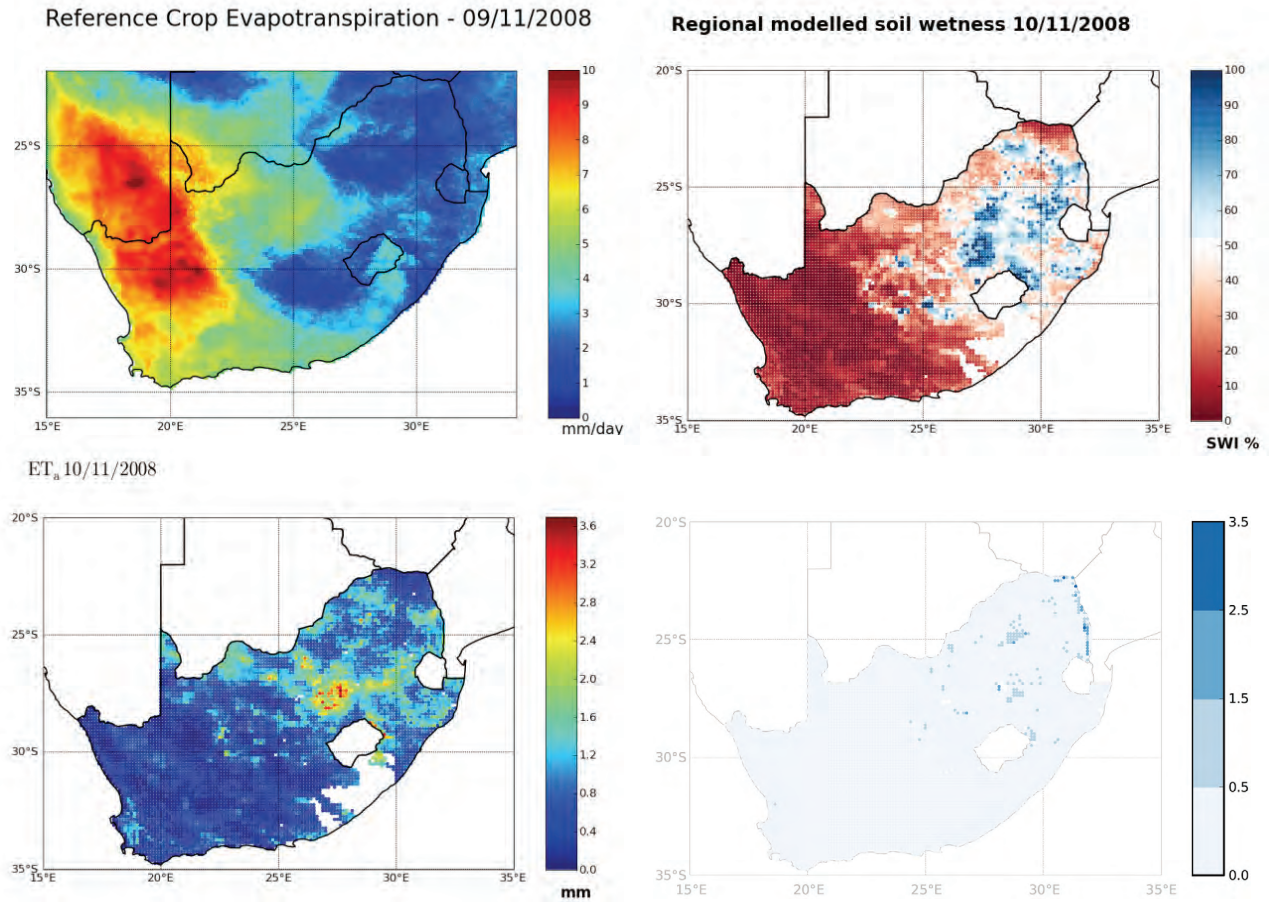


Figure 4.7 Eta sensitivity. The first three panels show the ET_0 , the current SSI state and the resulting ET_a on the same day in 2008. The fourth panel shows the ΔSSI map as a result of introducing the Green-Ampt infiltration module.

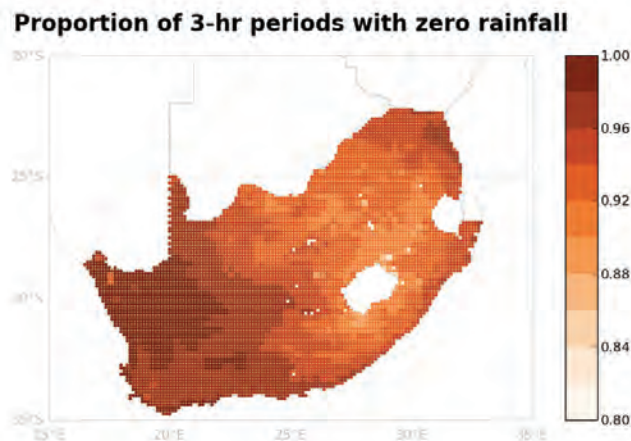


Figure 4.8. $P[0]$ at each TRMM site (25 km square) from observations during 1/8/08 to 30/6/00.

Because $P[0]$ is so high it implies that, to make a fair comparison, we should increase the SSI response to rainfall by a factor of about 2 and expect a resulting mean ΔSSI of the order of $\pm 8\%$. This observation led us to make a big effort to improve TRMM rainfall by conditioning the estimates of rainfall (via quantile transforms) to better represent what is measured on the ground. This was an

imperative, because TRMM rain turned out to be our most influential source of error, a variable upon which ET_o depends heavily. This issue will be dealt with in Section 5.

4.3. Comparisons of TRMM and DWA daily rainfall and a correction algorithm

The TRMM data, based on a new NASA algorithm, are available at 3 hour intervals in average values over 0.25° (~ 25 km) square blocks, as indicated by the size of the mesh in Figure 4.8. The information comes from a blend of orbiting radar estimates of rainfall augmented by Geostationary satellite scans.

The TRMM data have been downloaded daily from NASA for the whole country during the period 1 Oct 2008 to date. For the purposes of the comparison with raingauges, we have accumulated the TRMM data over days, to match the gauge rainfall read at 8 am. They have been used at 3-hour intervals in the derivation of Soil Moisture estimates using PyTOPKAPI during this period, as detailed in Chapter 4.

The purpose of the correction performed here is to assess the sensitivity of the SM estimates to the adjustment and hence determine whether, and by how much, routine adjustment of the TRMM rainfall will increase the quality of the Soil Moisture estimates. What resources are available which are appropriate for the task? As we did not obtain the SAWS data for the 21 months period of interest, we sourced DWA data and were kindly provided with 105 sets covering 1 Oct 2008 to 30 Jun 2010. Figure 4.9 gives their locations. The two green-ringed sites (stations A2E022 and A2E026) and the orange-ringed site (A6E005 to their Northeast) are discussed in Section 4.2.

The obtained DWA daily rainfall data tend to be available in places where water is impounded or on big rivers at gauging stations, sorted by DWA region (see Figure 4.10). They group quite nicely in the areas experiencing higher rainfall as seen in Figure 4.9.

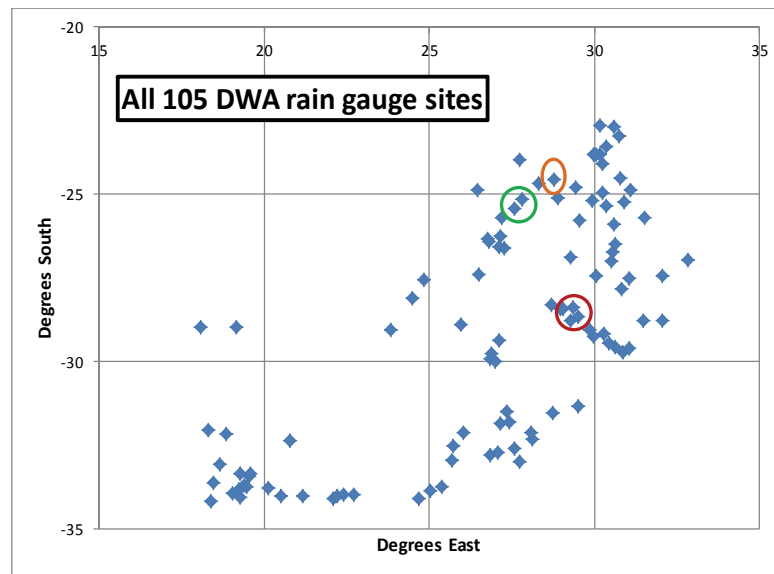


Figure 4.9. Locations of the 105 DWA rain gauges offered for this study. The red-ringed sites are in the Liebenbergsvlei to be referenced in the discussion on Figure 4.14. The two green-ringed sites (stations A2E022 and A2E026) and the orange-ringed site (A6E005 to their Northeast) are discussed in Section 4.3.2.



Figure 4.10. DWA's Drainage Regions, labelled.

The difficulties we faced are summarised as follows. The first problem was that the DWA data were few – only 73 series were able to be used out of the 105 offered because of missing or erroneous data records. In order to indicate how we weeded out the bad records, Figure 4.11 presents a pair of double mass plots between neighbouring pairs of gauges: one good, the other bad, from the same region in the upper row.

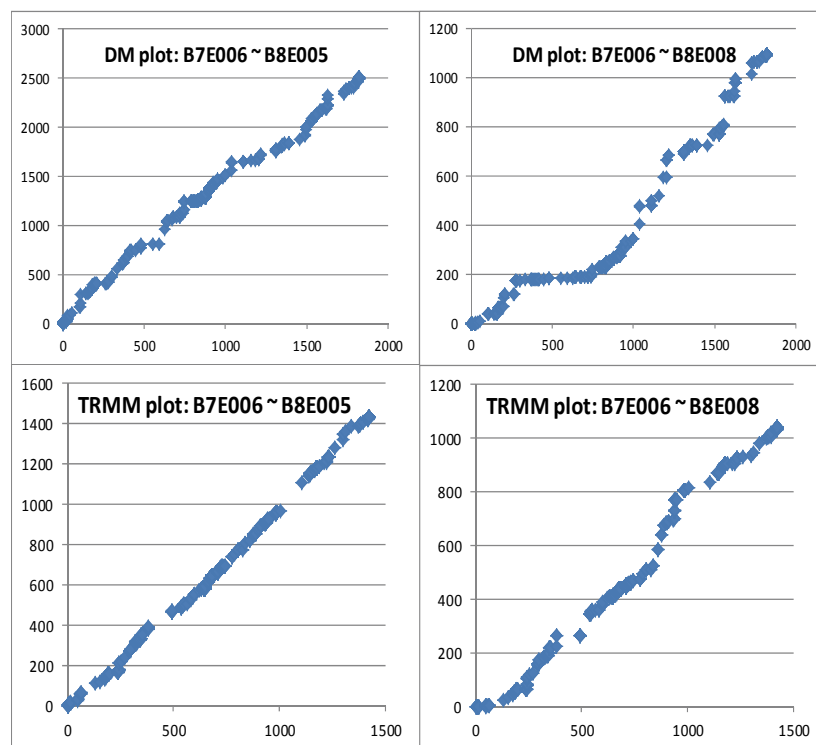


Figure 4.11. Two sets of double mass plots. Two pairs of three gauges from the same region (B7E006 plotted against B8E005 & B8E008), in the upper row (gauge~gauge). In the lower row, the data from the matching TRMM pixels containing the three gauges have been compared in contemporaneous double mass plots (TRMM~TRMM).

In the lower pairs of the figure, the data from the TRMM pixels containing the matching gauges have also been compared in double mass plots. Only gauge B8E008 indicates almost no rainfall in a 2 month period in the wet season, so is clearly in error. Therefore the fate of B8E008 (and others exhibiting the same behaviour) was exclusion from the analysis. Other techniques, such as outlier detection and rank correlation between neighbouring gauges were also used for quality control.

The second problem was that, because the gauge data are daily we are unable to downscale directly to the diurnal cycle, so we needed to use daily scaling adjustments to make the 8 prorated estimates of the 3 hour TRMM estimates.

The most important problem is that TRMM is relatively highly biased compared to the gauges. Figure 4.12 (drawn from the MAHYVA project report WRC 1747/1/10: Rouault et al., 2010) shows the diurnal variation of rainfall of TRMM3B42 (gauge-corrected by NASA) over the Liebenbergsvlei in the summer months the period during 1998-2007. Here an over-estimate by TRMM of the amount of rain is indicated when integrated over the full day.

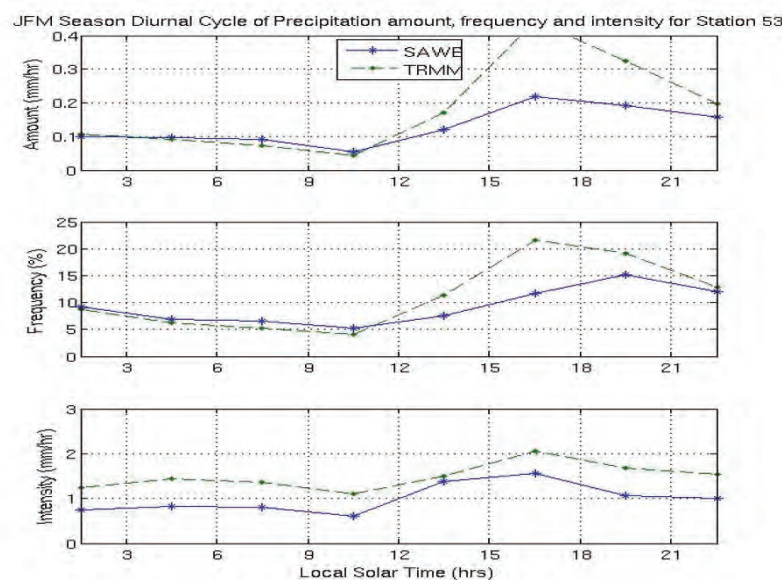


Figure 4.12. Mean diurnal cycle of precipitation amount, frequency and intensity for the 'Vlei for January February and March. [Figure A1.5a. in the MAHYVA project report WRC 1747/1/10; Rouault et al., 2010]

Our analysis shows (on a daily basis) much the same increase of TRMM versus gauge estimates for the station C8E008: see the left panel of Figure 4.13, which compares running accumulations of rainfall over the 2008-10 period for the two sets of instruments; the site of the gauges is red-ringed in Figure 4.9. In the panel on the right of Figure 4.13 is a Double Mass plot comparing the rainfall at gauges C8E009 against C8E008 (both at Sterkfontein Dam) which lie within a kilometre of each other. They both have good records and an almost 1:1 relationship. Clearly TRMM overestimates gauges at these sites, in concert with the observations in Figure 5.6.

We have no means of validating the 3-hourly diurnal estimates at this stage, without obtaining additional appropriate sub-daily rainfall data for the period since 2008-10-01. However, we can perform a comparison exercise on the downscaled daily TRMM accumulations, to compare with the daily gauge records. These comparisons appear later in this section.

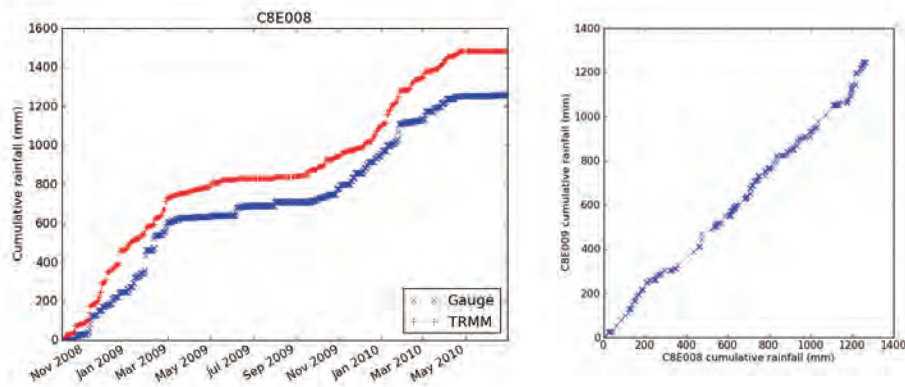


Figure 4.13. Left: A comparison of running accumulations of rainfall at station C8E008 and the TRMM pixel it lies in, over the 2008-10 period; the site of the gauge is red-ringed in Figure 4.9. TRMM estimates are routinely higher than gauge at this site. Right: A double mass plot of station C8E009 versus C8E008 (both at Sterkfontein Dam, lying within a kilometre of each other), showing excellent 1:1 similarity.

An anticipated difficulty (later confirmed by analysis) is that the mountainous coastal zone from the Southwest through to the Northeast of the country reflects high local variability of rainfall, particularly in the folded mountains of the South-western Cape Province. In addition, because the Geostationary satellites' rainfall algorithms involve primarily cloud-top temperatures to estimate rainfall, we thought it was likely that the coastal rainfall is underestimated because it is relatively 'warm rain'. The radar of the orbiting satellite should correct this but there are times when it skips certain regions in its programmed swath.

We were concerned about the quality of the transformation that results from the sparse DWA gauge set, especially in the dry interior (lack of water therefore no dams of size). However, from a pragmatic Soil Moisture (SM) modelling point of view, this lack of information in dry areas is not as important as is the availability of SM information in higher rainfall areas where agriculture, populations and therefore dangerous floods occur. There, the density of DWA rain gauges is greater, as indicated in Figure 4.9. Nevertheless this pilot investigation needs to be expanded.

The short period of contemporaneous TRMM and DWA data was an initial worry because of the likely paucity of rain-days to make the comparisons. However, comparisons made in relatively high rainfall regions of the country ($\text{MAP} > 500 \text{ mm/y}$) indicate that the frequency distribution functions (FDFs) of the gauges grouped in their DWA region show similar behaviour, once the poor ones have been teased out. In Figure 4.14 we compare the FDFs of a SAWS gauge and the average of three DWA gauges.

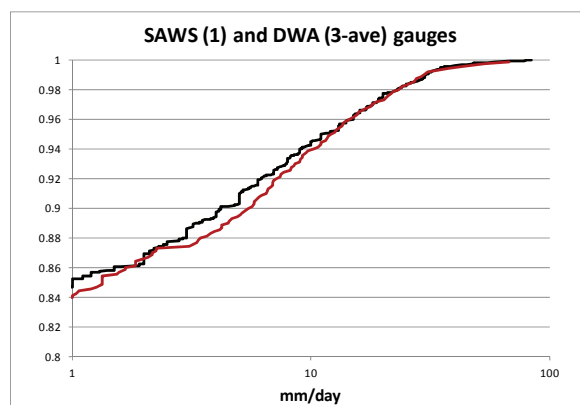


Figure 4.14. Comparison of FDFs of a SAWS gauge against three DWA gauges averaged, grouped in the red ring in Figure 4.9. The vertical axis is cumulative relative frequency.

The gauges analysed in Figure 4.14 are all near each other, to see if there was reasonable similarity between their distributions. The correspondence is good, even though the DWA gauge records cover the 2008-2010 period while the SAWS gauge (selected from the Lynch data-base cleaned up for project K5/1964) covers the period 1990-2000. We took this similarity as partial justification of our methodology.

4.3.1 TRMM adjustment.

The methodology used to conduct the TRMM adjustment follows. Because of the short records and the sparseness of the gauge data, after some experimentation as described in the preceding paragraph, it was decided to pool the DWA daily rainfall record within each Primary drainage region for the purpose of downscaling TRMM data to gauge values. The exception was Region D, which stretches from the source of the Orange River in Lesotho, to its mouth in the Atlantic Ocean (see Figure 4.10); here we separated out the single gauge in the West and pooled it with the West-coast set in Region E.

The DWA data-set was treated to remove those gauges which were inconsistent with the others; this was achieved by manually and visually comparing FDFs and Double-Mass (DM) plots of geographically contiguous data records, as indicated in Figures 4.12 and 4.13. Using the same methods, they were compared to the TRMM daily constructed records, where only TRMM pixels containing gauges were selected to compare with the gauges contained in the pixel concerned as shown in Figure 4.13.

The decision to pool the gauge and matching TRMM data to produce blended FDFs by Drainage region was predicated on the following ideas:

- the FDFs of the individual gauges on the one hand and TRMM on the other compared well enough, especially in the hinterland, even though they do not do so well in the mountainous coastal zones, as expected; we cannot solve this problem with few gauges.
- the blended rankings of the rainfall amounts would match well (based on the results of the DM plots) and preserve the tendency for drier areas in a region to reflect matching lower gauge and TRMM estimates and vice versa. This would only be so if TRMM was doing well in a spatial sense.
- the exercise of downscaling TRMM estimates using the pooled data of gauges directly (and not their spatial averages) is founded on the reason that we need rainfall estimates at the 1 km pixel scale for PyTOPKAPI responses, rather than at the Drainage region size. We need the FDFs at near PyTOPKAPI pixel scale to evaluate the sensitivity of the model to TRMM correction.

4.3.2 Use of the output from WRC K5/305 to assist in TRMM downscaling

The stated objective of the WRC report by McNeill et al. (1994) was:

“The main objective of this project has been to produce estimates of the parameters of the daily rainfall model of Zucchini and Adamson (1984) for sites throughout South Africa at which there is little or no rainfall data available, thereby making it possible to use the model to generate artificial rainfall sequences and study rainfall characteristics at any given location or over any given area in South Africa.”

The map of the sites, taken from Figure 1 of the report by McNeill et al. (1964), appears in Figure 4.15 following, which shows the density of the gauges used in that WRC study. We were able to find a SAWS station from this set which was at most 78 km away from any DWA station, with 70% of them within 10 km range. The idea was to generate 10 years of simulations at SAWS sites selected from the report which were close to the DWA stations in order to see if we could add information to our short,

sparse DWA gauge set. We anticipated that these would give us useful FDFs based on stations with records longer than 2 decades in the past which had been carefully screened for error.



Figure 1: Stations with at least 20 years of data.

Figure 4.15. Station sites from WRC K5/305: location of 5070 stations with at least 20 years of data. [McNeill et al., 1994.]

We were able to obtain the electronic files for the simulation of the 10 years of daily data based on the WRC project K5/305 package, thanks to the helpful people at BEEH, who rescued it for us from the remnants of the CCWR archive. From these sets of simulated daily values based on pre-1992 data, we were indeed able to construct FDFs at sites near the DWA locations. Some appear below in Figure 4.16. However, on comparison there was a large difference between the FDFs when measured using a 2-sample Kolmogorov-Smirnov test. Specifically, for the quantities concerned, (22 months for DWA and 10 years for SAWS) the maximum measure D-statistic to allow similarity between the FDFs, turned out to be 0.059 and 0.071 for significance levels of 5% and 1% respectively.

A sample of comparisons appears in Figure 4.16, where the D-values in the two images in the top row, where we respectively compare first DWA gauges and then SAWS simulations, are 0.038 and 0.011, both well within the bounds, indicating close similarity. However, for the bottom row the distances between DWA and SAWS simulations are 0.076 and 0.057, which indicate that the FDFs are respectively similar with probabilities of only 0.004 and 0.052, indicating they are not only visually (beware the log-log plot and the switch of axes!) but also statistically different.

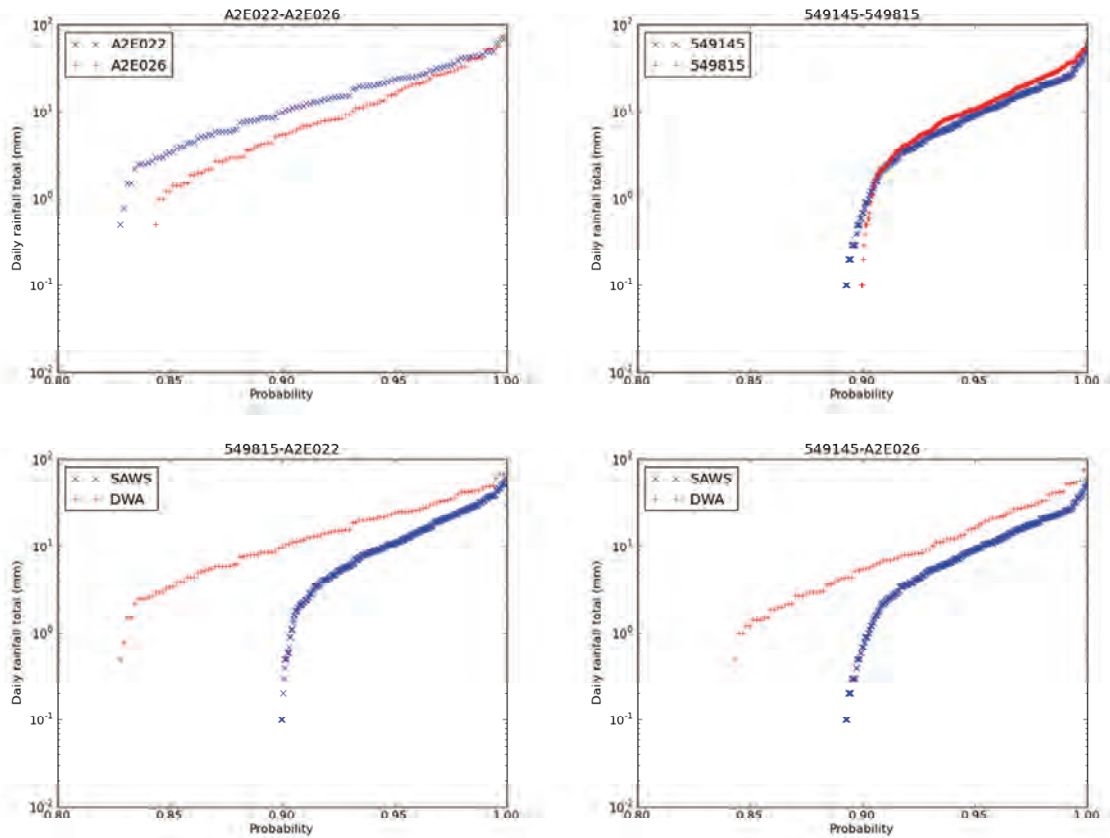


Figure 4.16. Comparison between DWA data for 1-10-2008 to 30-8-2010 against WRC/305 SAWS simulations for 10 years based on data prior to 1992. Top left FDFs of DWA gauges A2E022 and A2E026 (appearing in the green ring in Figure 8) plotted coaxially. Top right, the 10-year simulations from K5/305 for the nearest SAWS gauges to the DWA ones. Both pairs are comfortably similar. Below, left and right, the DWA and SAWS FDFs from above plotted against their neighbours.

As a further comparison, when we plot the FDF of DWA gauge A6E005 (orange ringed in Figure 4.9) versus its SAWS simulations, it behaves nicely ($D = 0.023$ with $P = 0.933$) as shown in the top left panel of Figure 4.17. But the joy is short-lived.

In the other 3 panels of Figure 4.17 (top right and bottom row) we plot contemporaneous accumulations of rainfall of three DWA gauges (red and green-ringed in Figure 4.9) against their matching TRMM pixel values over the 22-month period. We emphasize that these plots are gauge versus TRMM (not SAWS as in Figure 14). A6E005 (top right) is drier than the matching TRMM record, whereas A2E022 and A2E026 (bottom row) are wetter than their corresponding TRMM estimates. This put us in the following quandary. Do we use the long historical data SAWS simulations or do we go with the short TRMM sequences?

The upshot was that we decided to stay with the pooled DWA data to perform the Q-Q transforms of TRMM at this juncture and not use the WRC K5/305 simulations to create the target FDFs.

An additional comment is appropriate here. From the three gauge-TRMM FDF comparisons of Figures 4.16 and 4.17 (not necessarily for others), it is clear that the period 2008-10 is wetter (both with regard to amount and $P[0]$) than before 1992 in this area. This observation may be useful in another context.

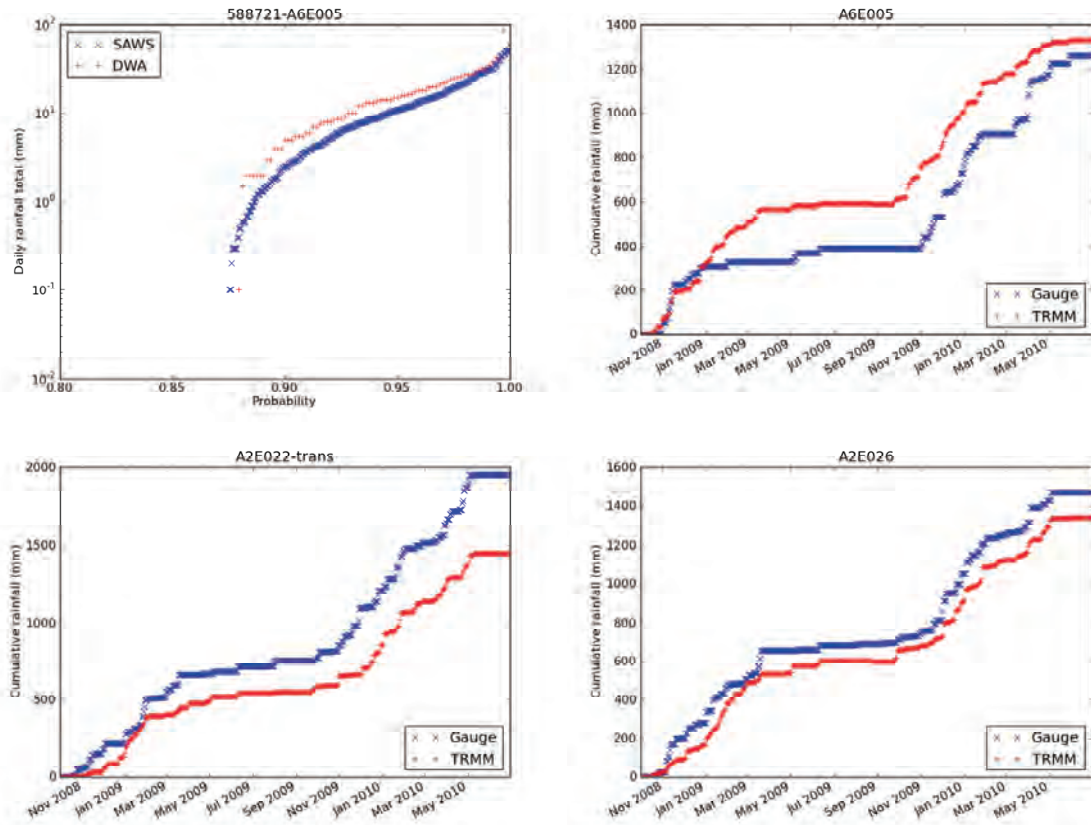


Figure 4.17. Plot of DWA gauge A6E005 (orange ringed in Figure 4.9) versus its nearest neighbour SAWS simulations in the top left panel. Remaining 3 panels: contemporaneous accumulations of rainfall for each of the three DWA gauges against the corresponding TRMM (not SAWS) estimates over the 22-month period.

4.3.3 The way forward

As explained in Section 4.3.2, our final choice of strategy was to pool the similar DWA data in each of the regions and use that as the means of a quantile-quantile (Q-Q) transform of the TRMM estimates at each of their locations within the region. We acknowledge that this procedure is a compromise, but we need the overall picture of whether the TRMM data are close enough to the gauged data for us to use the product as it is, or else put in the future effort to routinely improve TRMM rainfall estimation by Q-Q transforms.

An example of gauge and TRMM data before and after transformation is given in the pair of images comprising Figure 4.18. The comparisons are for gauge G1E006 in the Western Cape and its matching TRMM pixel. Although the marginal distributions are comparable (given later in Figure 4.20) and most of the events match in a rough way, there is unfortunately no way of saving the comparatively low TRMM estimates in May-June 2009. However, consideration has to be given to the fact that the gauges are point estimates and the TRMM estimates are averages over 25 km squares.

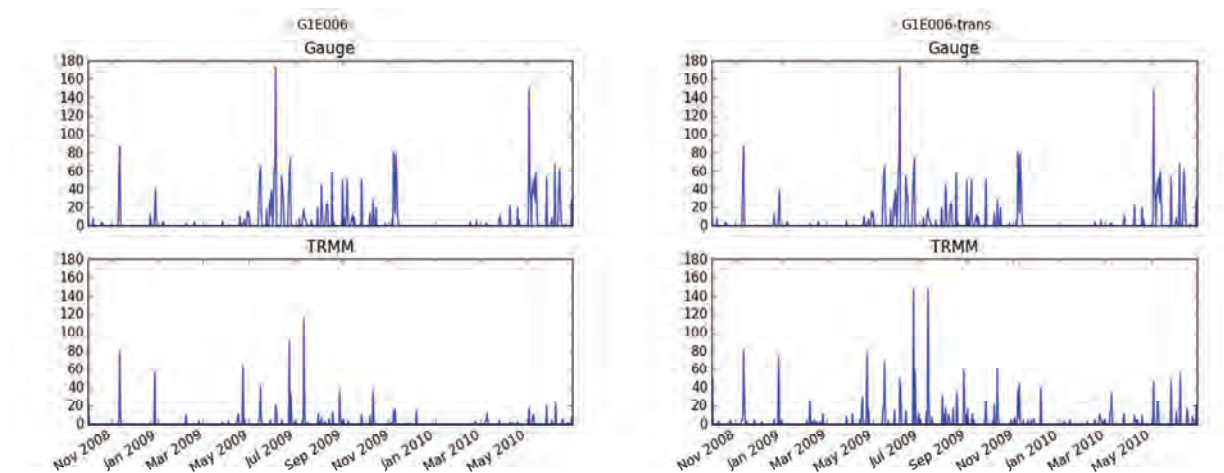


Figure 4.18. Transformation of a TRMM time series using pooled DWA data from region G. The data from the TRMM pixel containing gauge G1E006 is compared before and after Q-Q transformation, respectively in the left and right pairs of images.

In the event, the TRMM Q-Q transformation was conducted on all regions where there were “good” DWA gauges, for the study period. The resulting change in SSI at each PyTOPKAPI model site was calculated in the same way as the other adjustments were done, keeping the remaining parameters the same as for the control run. The result is presented in Figure 4.19, where the discontinuity between the DWA hydrological regions (shown in Figure 4.10) is evident. The white areas are regions where no appropriate DWA data were available.

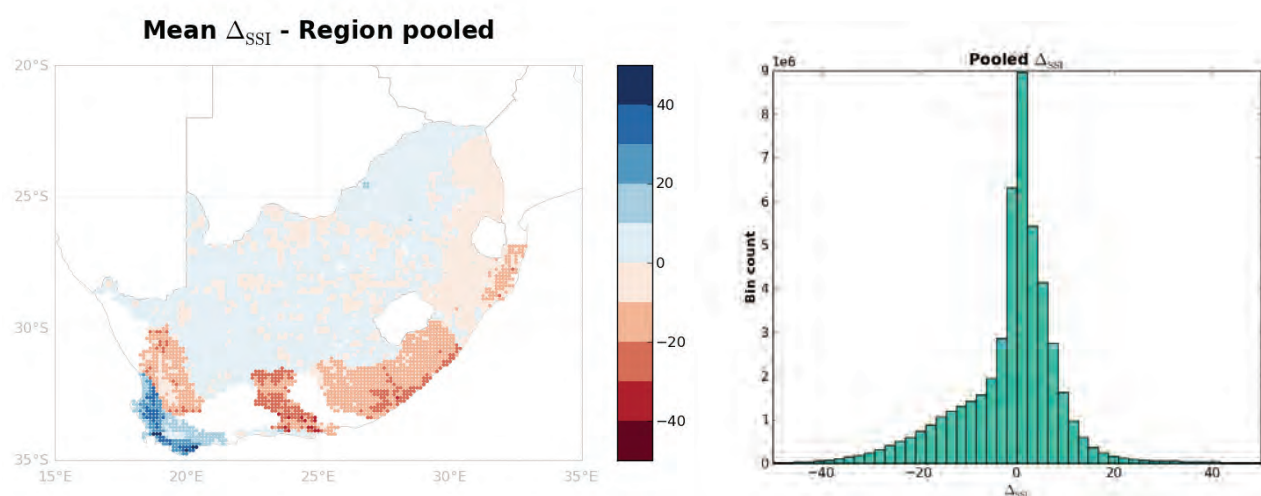


Figure 4.19. Mean change of SSI (%) due to TRMM correction using DWA gauges at each PyTOPKAPI site. This is the result of the TRMM transformation using the DWA gauges pooled by region (see Figure 10 for labelling of the drainage regions). Left panel: spatial distribution of cell averages; right panel: histogram of pooled 3-hourly values of change.

The discontinuities appearing in the left panel of Figure 4.19 at the Region boundaries are caused by pooling the responses of the 73 DWA gauges, but the overall picture is clear: TRMM estimates definitely need to be bias corrected, because the range of the effect on the SSI is $\pm 40\%$, with positive and negative means of 5.5% and -8.3%. These means were calculated from the data forming the green

histogram in the right panel and are close to the range $\pm 8\%$ that we expected based on the discussion following Figure 4.8.

Another lesson learned here is that TRMM estimates compare with DWA values much better over a 'smooth' interior than a rough, coastal terrain. There is a clear difference between behaviour over the hinterland (not much variation over the summer rainfall regions) and the coastal zone. Along the Southeast coast, it seems that TRMM overestimates the rainfall, whereas in the Western Cape the opposite occurs and TRMM is too dry in those parts.

In more detail, the corrections to TRMM cause ΔSSI in drainage regions A, B, C, D and even K to fluctuate either side of zero as do U, V, the upper regions of W and X. The coastal zone from Cape Agulhas to the Mhlatuze River: regions L, M, R, S, T and the lower lying areas of W have all experienced drier SSI estimates after TRMM has been bias corrected. The change in SSI in the Western Cape region G and to a lesser extent H move the other way, because the TRMM estimates were far too dry compared to the pooled records of the raingauges.

As verification of the method, the 4 panels in Figure 4.20 show some gentle (A) and severe (G) Q-Q transforms of TRMM using the pooled DWA data, when compared to the individual gauges. The result is in part an exoneration of the decision to pool the gauge data by region and then use these to adjust the individual TRMM pixels. The bad one here is G2E001, whose higher quantile values have been increased beyond the observed maximum of 30 mm to about 80 mm, whereas G1E006 is much better captured, as shown in Figure 4.18 above.

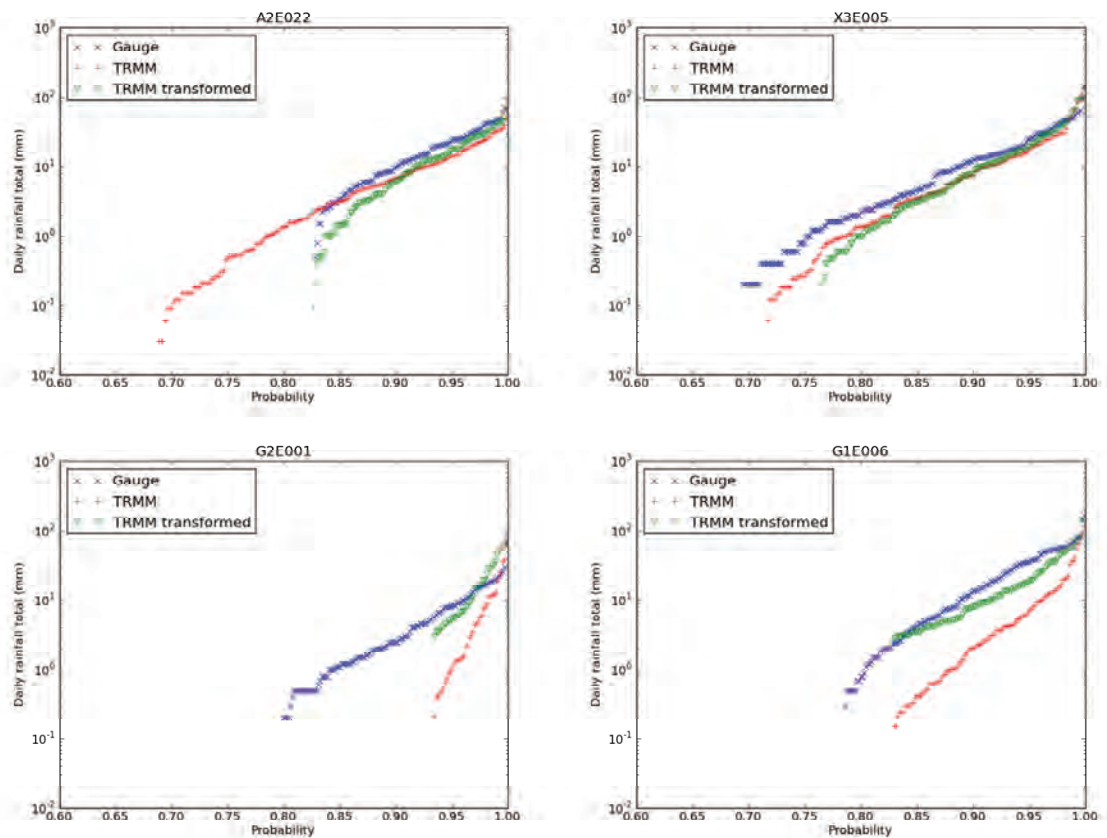


Figure 4.20. These FDFs show the result of the TRMM Q-Q transformations using the pooled data for the regions. Region A contains 6 and region G contains 4 DWA gauges pooled in their respective regions to perform the transforms.

4.4. Summary

This chapter considered the problem of improving those forcing variables of the PyTOPKAPI model, which have an influence on the Soil Moisture (SM) response of the 7200 sites on a grid across South Africa.

The work was done in two stages: (i) assess the sensitivity of SM to variations in the parameters and inputs (ii) correct those data with the greatest influence on the Soil Saturation Index (SSI), the SM surrogate.

Not surprisingly, it turned out that the SSI was most sensitive to the rainfall input, then to the soil parameterization and much less to actual evapotranspiration (ET_a). We felt it important to find what steps were necessary for adjusting both the ET_a estimation and the remote sensing estimates we obtain from NASA's TRMM 3B42RT 3-hour rainfall product.

The Chapter was divided into the following main components in Sections 4.3 and 4.4.

- (i) sensitivity of SSI to the following variables: soil parameters (depth, effective porosity and conductivity); rainfall depth; ET_a and
- (ii) the evaluation and adjustment of the TRMM rainfall input.

From the comparative sensitivity analysis in Section 4.3, it turned out that the TRMM rainfall estimates are the most influential of all the variables affecting SSI.

We had available 638 days of data with which to cross-compare gauges with TRMM daily rainfall estimates. Because of the small number of gauges available, we had to pool them spatially to achieve what was done here.

Using these, the work reported in Section 4.4 convinced us that the TRMM product needs significant quantile-quantile transformation to correctly estimate rainfall. This is especially so in the coastal zone of the country, as it has steep and in many places mountainous terrain. The comparatively flat hinterland, by contrast, appears to be relatively well served by the TRMM rainfall product.

It remains to build on these results to produce a sound method for routinely performing quantile-quantile transforms of TRMM to ensure a sound rainfall product with which to feed PyTOPKAPI. To do this, we would need to employ a denser data-base of up-to-date raingauge records. Nevertheless, with the resources available, the study has exposed the fact that TRMM, more than any other variable associated with Soil Moisture, needs correction to add value to hydrometeorological modelling in RSA. As importantly, we have provided the prototype of the methodology for making these corrections in an effective and routine way.

5: SSI Estimates of HYLARSMET on SAFFG Catchments

Preamble

The purpose of the work reported in this chapter was to adapt the PyTOPKAPI package to be applied to any chosen catchment in South Africa, with special emphasis on the SAFFG catchments of the SAWS.

We demonstrate here that PyTOPKAPI can model the SAFFG catchments, in extent, drainage and soil wetness, noting that the SAFFG catchments have areas of the order of tens of square kilometers, typically being small urban and peri-urban catchments. This chapter tells the story of how this match was made. The synergy was facilitated by our being partners with the SAWS team, headed by Mr Eugene Poolman, in the WRC project K5-2068 entitled: 'Improvement of early preparedness and early warning systems for extreme climatic events flood warning'. That project involves the installation and utilisation of the FFG system devised by Prof Kostas (Konstantine) Georgakakos at the Hydrologic Research Center, San Diego, California.

In passing, the link between Pegram and Georgakakos goes back to 1997, when Pegram presented his first results on the WRC-funded 'String of Beads' radar-rainfall model at the Radar Hydrology Symposium in San Diego. In that visit, Pegram was inspired by NOAA's Flash Flood Forecasting centre in California where meteorologists and hydrologists worked in the same room using the same data. This, in contrast to the Flood Warning Centre near Reading, where the Flood Forecasters are separate from (and have a subset of) the source of the remote sensing information they have to purchase from Met Office. As a result Pegram called a meeting of WRC, SAWS, and DWA in Bethlehem on May 15 2005, where the idea of the Flood Forecasting system was born. The SAFFG is the fruit of that meeting, encouraged by the work reported in a WRC contract (Pegram et al., 2007).

Returning to the matter at hand, one of the main reasons for the existence of the current research project was that the members of the review committee of the prior WRC project K5/1683 felt that that our preliminary product was of great potential value. To realise this potential, the research team was encouraged to extend the research in a follow-on phase in order to make substantial improvements to the earlier TOPKAPI prototype; hence this HYLARSMET study. In particular, areas to be addressed include [excerpted from the contract document of this project]:

- Improvements to input data. Current 3-hour rainfall spatial estimates obtained via the TRMM satellite are evidently biased. Furthermore, the soil data currently utilized are limited and out-dated.
- Comparisons with other sources of information. There is an urgent imperative to ensure thorough validation in order to build confidence among user groups.
- Assessing the effect of scale of the product. Determine whether 7000 Land Surface Model (LSM) sites provide sufficient information about Soil Moisture in a country of 1.2 million square kilometres. To model the whole is computationally demanding; in addition, the remote sensing information we currently have access to has a spatial scale of from 3 km to 25 km square.

We want to forge the link between the isolated LSM sites at 12 km spacing with hydrological response of: (i) small catchments modelled at 1 km spacing and (ii) large catchments modelled with 8 to 12 subcatchments.

The potential value of the outputs from this work is also outlined in this contract's document in the following passage. "Through the members of the reference group of K5/1683 and the Soil Water

Workshop hosted by the WRC in 2007, several researchers/institutions have expressed a desire to make use of an improved product. They include: ARC(ISCW) for crop modelling; SAWS for their Flash Flood Guidance System and possible involvement in the AMESD initiative of EUMETSAT to provide SM estimates to SADC countries.” [We have already been feeding ARC with monthly SM outputs; this work forms the links with SAWS’ FFG system; we have yet to go beyond our country’s boundaries – that may come in the future. Dr Sinclair attended the first steering committee meeting of K5-2068, and established an agreement with Mr Poolman to obtain time series of Soil Moisture estimates on selected SAFFG catchments under our partner agreement under the contract.]

In summary, the primary goal of the work reported in this chapter was to extend the work done to model country-wide Soil Saturation Index (SSI) at isolated locations, by setting up hydrologically consistent PyTOPKAPI models at the scale of the SAWS SAFFG catchments. This approach allows detailed modelling of interconnected and contiguous 1 km² cells at fine scale, as an alternative to modelling isolated 1 km² cells at a grid spacing of approximately 12x12 km (as done for the country-wide Soil Moisture assessments). The eventual aim is to assess the role of lateral transfers of water between cells in the modelled SSI.

The work done to achieve this goal consisted of two components:

1. develop tools to (partially) automate the process of setting up a PyTOPKAPI model in new catchments, so that fine scale modelling of many SAFFG catchments becomes feasible, and
2. test these tools by using them to do detailed modelling on a selection of SAFFG catchments.

Worth noting at this juncture is that this novel automation reduces to hours and days what was preciously only achieved in months and years by hand, when we originally set up TOPKAPI for the Liebenbergsvlei. In section 5.2 we revisit the ‘Vlei using the new techniques to make sure we are on the right track – we found that we are!

Each of these components is outlined in the sections that follow.

5.1. Automating the PyTOPKAPI Model Setup on New Catchments

In order to model a catchment using PyTOPKAPI, it is necessary for the user to prepare a parameter file that describes the catchments physical properties on a regular grid with the desired spatial resolution. This work needs to be carried out by the model user with whatever tools and spatial information they have at their disposal – typically using a GIS package to collate and manage the relevant information. We developed two toolsets to assist model users to streamline this process which *de facto* achieves the primary goal of the work reported in this chapter.

First, we carefully upgraded and streamlined the existing helper scripts in the PyTOPKAPI package to cater for the sophisticated user who is able to carry out the work of delineating their catchment using GIS, or other tools with which they are familiar. One of the new tools is a Python function in the PyTOPKAPI package that produces a valid and properly formatted PyTOPKAPI parameter file when provided with the required set of geo-referenced raster data files and some configuration parameters. The user needs to set up the locations of the files in the <...> brackets in the listing below, to tell PyTOPKAPI where to find the information it needs to run the model. This non-trivial innovation has considerably streamlined the setting up procedure.

The listing follows.

```

[raster_files]
dem_fname = <path to DEM file>
mask_fname = <path to catchment mask file>
soil_depth_fname = <path to soil depth file>
conductivity_fname = <path to saturated conductivity file>
hillslope_fname = <path to hill slope file>
sat_moisture_content_fname = <path to saturated moisture content file>
resid_moisture_content_fname = <path to residual moisture content file>
bubbling_pressure_fname = <path to bubbling pressure file>
pore_size_dist_fname = <path to pore size index file>
overland_manning_fname = <path to overland Manning roughness file>
channel_network_fname = <path to channel network file>
flowdir_fname = <path to flow direction file>
flowdir_source = <source of flowdir file. Can be 'GRASS' or 'ARCGIS'>

[output]
param_fname = <path to output parameter file>

[numerical_values]
pVs_t0 = <initial percent saturation of soil stores>
Vo_t0 = <initial volume of overland stores>
Qc_t0 = <initial flow rate in channels>
Kc = <crop factor, currently this must be set to 1.>

```

The tool takes care of all of the details that specify inter-connection between cells and the general model set-up and guarantees the user that a valid parameter file will be produced. However, producing the required input rasters for a catchment is a non-trivial task, so we developed a second tool to automate the process of delineating and analysing a catchment using standard GIS tools.

We elected to develop this second tool using the free and open-source package GRASS GIS because we believe that it's important to avoid requiring potential model users to purchase expensive proprietary GIS packages in order to take advantage of our tool-set. Additionally, using open-source tools makes it easier for the procedure to be critically evaluated and peer reviewed, because it is repeatable. The remainder of this section outlines procedure we have automated and then discusses two technical aspects that need special attention (filling of sinks in DEMs and computing the Strahler stream order to define the Manning streamflow roughness parameter in PyTOPKAPI).

The procedure to produce the set of raster files needed to generate a PyTOPKAPI parameter file is outlined in figures 5.1 – 5.7 and the supporting text; the figures are likely to convey more information than the text. The first requirement is that the GRASS GIS location contains an elevation model and other base maps for the region where the target catchment is situated (for this project we used the 1 km² country-wide base maps in chapter 2, but a similar set of data at an alternative spatial resolution could be used instead). The catchment modelled by PyTOPKAPI is defined by the properties of the DEM, so to ensure we have fully captured the extent of the catchment, an initial boundary is required (see Figure 5.1), obtained from WR2005 or SAFFG files. PyTOPKAPI now possesses code that combines the GIS tools, in an automated way, to carry out the steps to define the catchment on the basis of the DEM, as outlined in Figures 5.1 to 5.6 that follow.

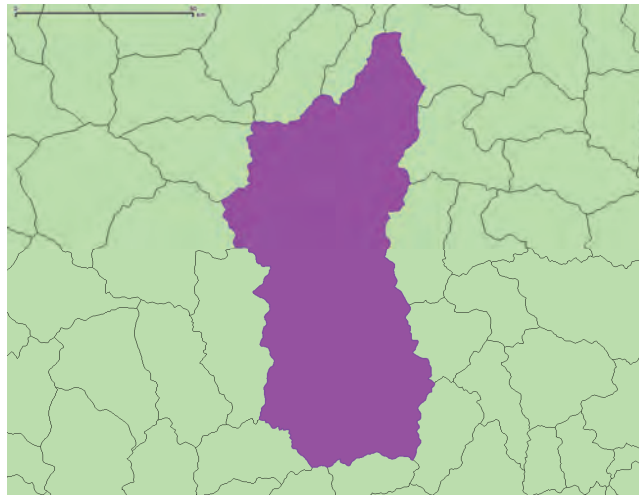


Fig 5.1: A vector polygon representing the portion of the Liebenbergsvlei catchment that we want to model using PyTOPKAPI. The green background shows WR2005 quaternary catchments, the purple overlay polygon is a concatenation of the sub-set of those quaternaries that make up the Liebenbergsvlei. [The bar at the top of the figure is scaled to 50 km]

The processing tool we have introduced uses functionality in the GIS to produce a 5 km buffer around the predefined catchment boundary (Figure 5.2). This is necessary because the catchment defined from the DEM will not necessarily match the boundary given in Figure 5.1 in an exact way.

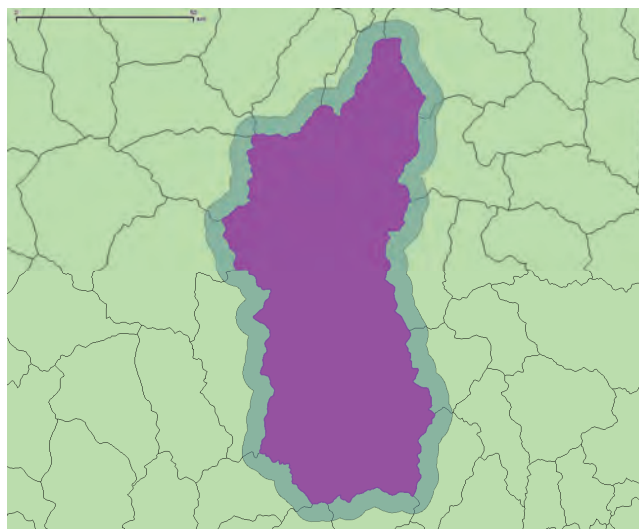


Fig 5.2: A 5 km buffer around the initial boundary provided. The buffer is indicated by the wide grey outline. [The bar at the top of the figure is scaled to 50 km]

The buffering accounts for this possible mismatch of the boundaries by ensuring that a sufficiently large portion of the DEM is examined in the vicinity of the catchment to be modelled. The external of the buffered boundary is used to cut out a portion of the base DEM for analysis.

The hydrological analysis tools provided by GRASS GIS can produce sensible results even if there are inconsistencies in the DEM. The most common inconsistency is a sink, where a cell or a collection of cells is lower than all of its neighbours. PyTOPKAPI assumes that water from all cells in a catchment will eventually drain down to a single cell at the catchment outlet, so DEM sinks must be removed, despite the fact that GRASS can provide useful catchment information from flawed DEMs. Figure 5.3 show the

progress of the automated sink-filling procedure for the Liebenbergsvlei. In the first panel (based on the SRTM, which stands for Shuttle Radar Tomography Mission) multiple sinks are identified (this is not uncommon) and two passes of the sink filling algorithm are required before all of the sinks are removed. The fourth image is the cropped basin.

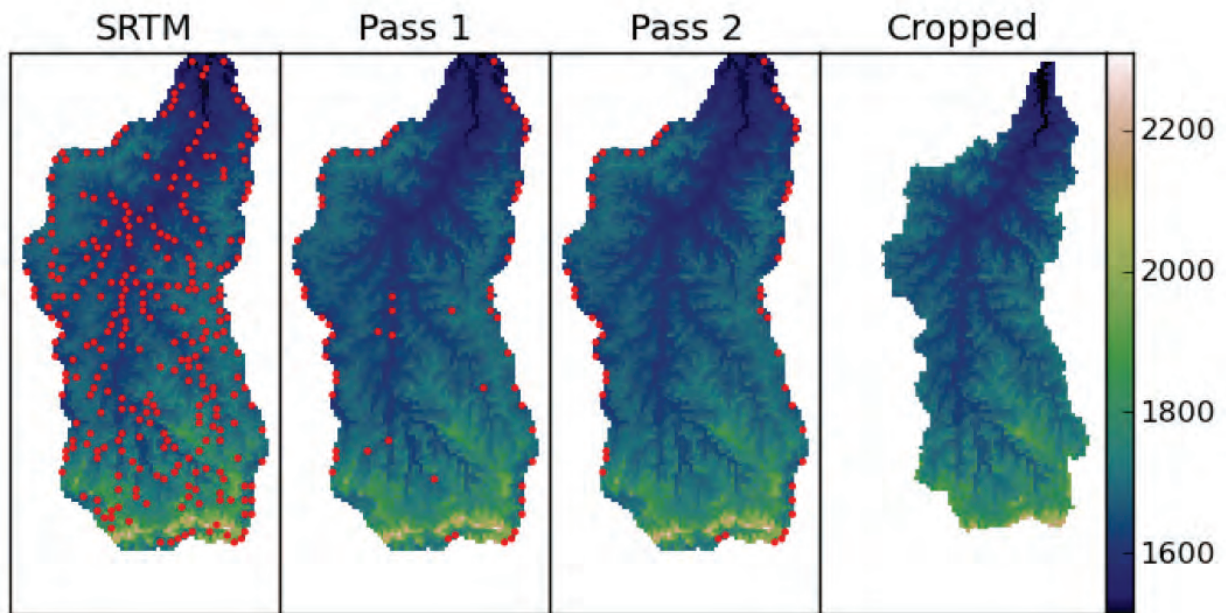


Fig 5.3: Demonstration of the automated sink filling routine for the Liebenbergsvlei DEM. The colour-bar on the right describes the elevation in metres a.s.l and the red dots show the locations of sinks at each stage in the process. The sinks on the border are ignored by the filling routine as they cannot be corrected without data in every cell surrounding the sink. The cropped catchment in the fourth panel is the region defined by the flow direction map, and includes all cells flowing into the outlet cell.

Once all sinks have been removed, the hydrological analysis tools in GRASS are employed to obtain (i) a flow-direction raster describing the interconnectivity of the cells as well as (ii) a stream network raster. These are illustrated in figures 5.4, 5.5 and 5.6.

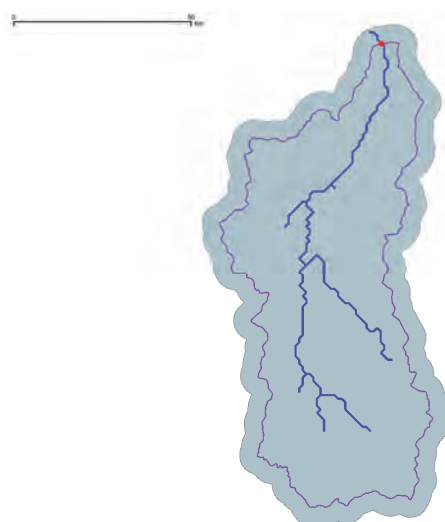
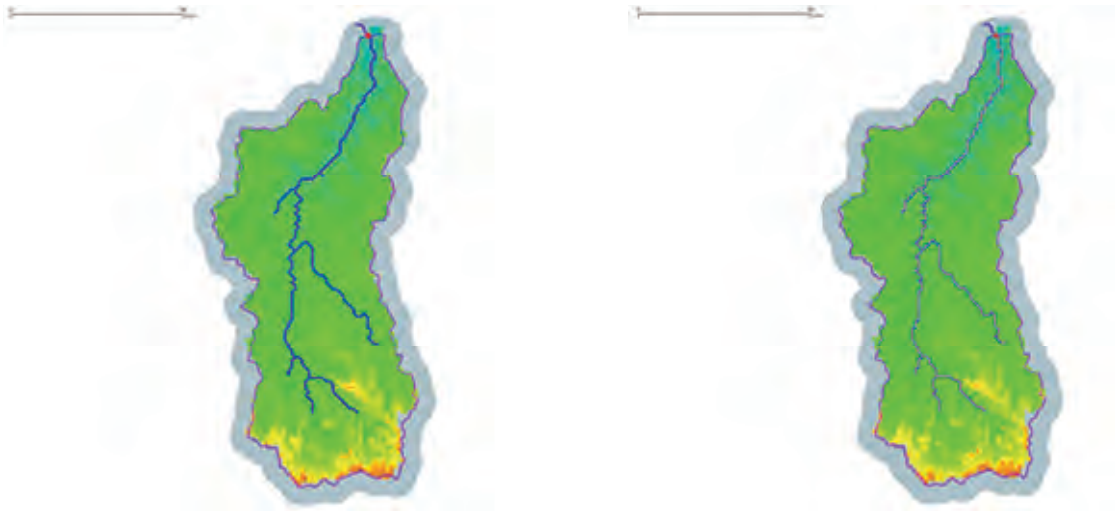


Fig 5.4: Vector version of the stream network obtained from DEM analysis (blue line). The red dot shows the catchment outlet obtained by calculating the position of the intersection of the stream network and catchment boundary. [The bar at the top of the figure is scaled to 50 km]



Figs 5.5 & 5.6: Left: The portion of the DEM extracted by computing upslope cells from the catchment outlet. Right: Raster version of the channel network, upstream of the catchment outlet.

The catchment outlet is also identified as being the cell where the stream network intersects the original catchment boundary (red dot in figures 5.4, 5.5 & 5.6). Once the outlet cell has been identified, the GRASS tools are used to follow the flow-direction information upstream until a watershed is reached, this process defines the extent of the catchment according to the DEM properties. Figure 5.5 shows the Liebenbergsvlei catchment extracted from the DEM, it matches very well (but not exactly) with the WR2005 based boundary that was initially provided to the processing tool in vector form.

Once the catchment has been delineated, the set of raster data files are produced to serve as input to the PyTOPKAPI parameter file generation tool. The entire process is carried out using a Python script, which controls the GRASS GIS tools, and ensures that the process is repeatable on different catchments. All that is required to process a new catchment is the vector definition of its boundary.

Figures 5.7 and 5.8 Give examples of model parameters obtained from two different sources, manipulated by GRASS: Surface Slope and Soil Depth



Figs 5.7 & 5.8: Left: Example of model parameters extracted. Surface slopes from the DEM. Right: Example of model parameters extracted. Soil depths from the Atlas

One part of the PyTOPKAPI parameter file generation process requires determining the channel roughness co-efficient based on the Strahler stream order for the given channel reach. We coded up an implementation of the algorithm described by Gleyzer et al. (2004) and this has now been made a part of the PyTOPKAPI package. Figure 5.9 shows the stream order computed on a synthetic stream network used to test the algorithm adapted for use in the PyTOPKAPI suite.

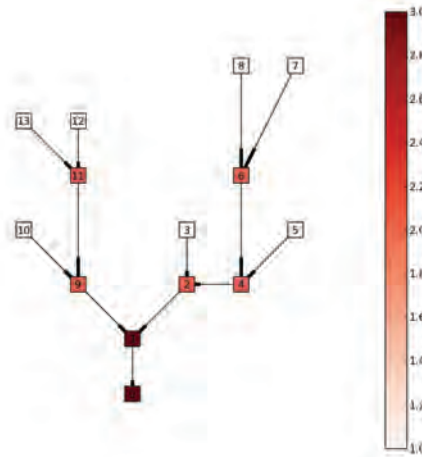


Fig 5.9: A directed graph representing a simple stream network. The numbered squares are the graph nodes and the lines joining the nodes show direction using a thickened end as an arrow head. The calculated Strahler order is show by each node's colour.

Figures 5.10 and 5.11 show the results obtained for the stream network extracted on the Liebenbergsvlei, using the algorithm displayed in Figure 5.9.

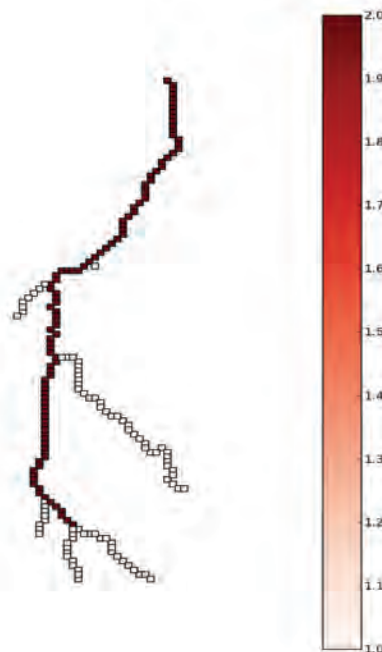


Fig 5.10: A directed graph representing the stream network extracted from the Liebenbergsvlei. The 1 km squares are the DEM graph nodes with Strahler order shown by their colour.

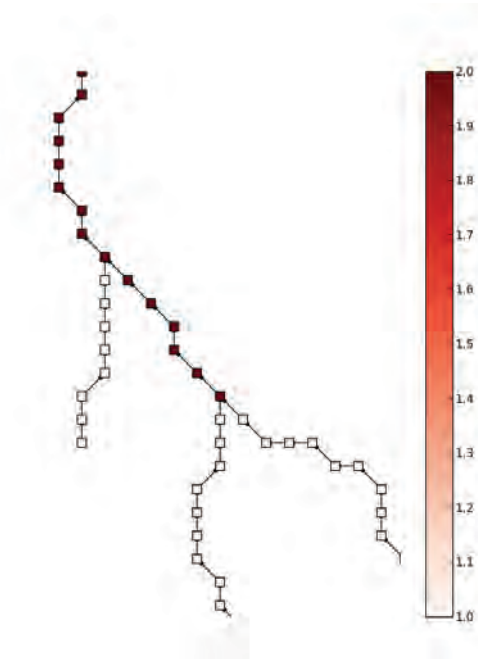


Fig 5.11: A close up showing the lower part of figure 5.10. Two first order streams join to form a second order stream, which does not change order when joined by another first order stream.

5.2 Soil Moisture estimates on SAFFG Catchments

The figures in this section provide an outline of the process used to model selected SAFFG catchments. In addition, they show some initial results. On advice from Eugene Poolman of SAWS, we selected four WR2005 tertiary catchments in close proximity to the Irene weather radar. We set up PyTOPKAPI models of each of these tertiary catchments using the tools described in the previous section and established rainfall and evapotranspiration forcing datasets for the catchments for the period 2008/08/01 – 2011/01/01. Simulations for this period were then run for each catchment and the average Soil Saturation Index (SSI) in the relevant SAFFG catchments was calculated for each 3 hour time-step of the simulations. Figures 5.12 to 5.17 deal with the location, definition and setting up of the catchments for simulation. Figures 5.18 to 5.21 summarize the simulation results that we have analysed to date.

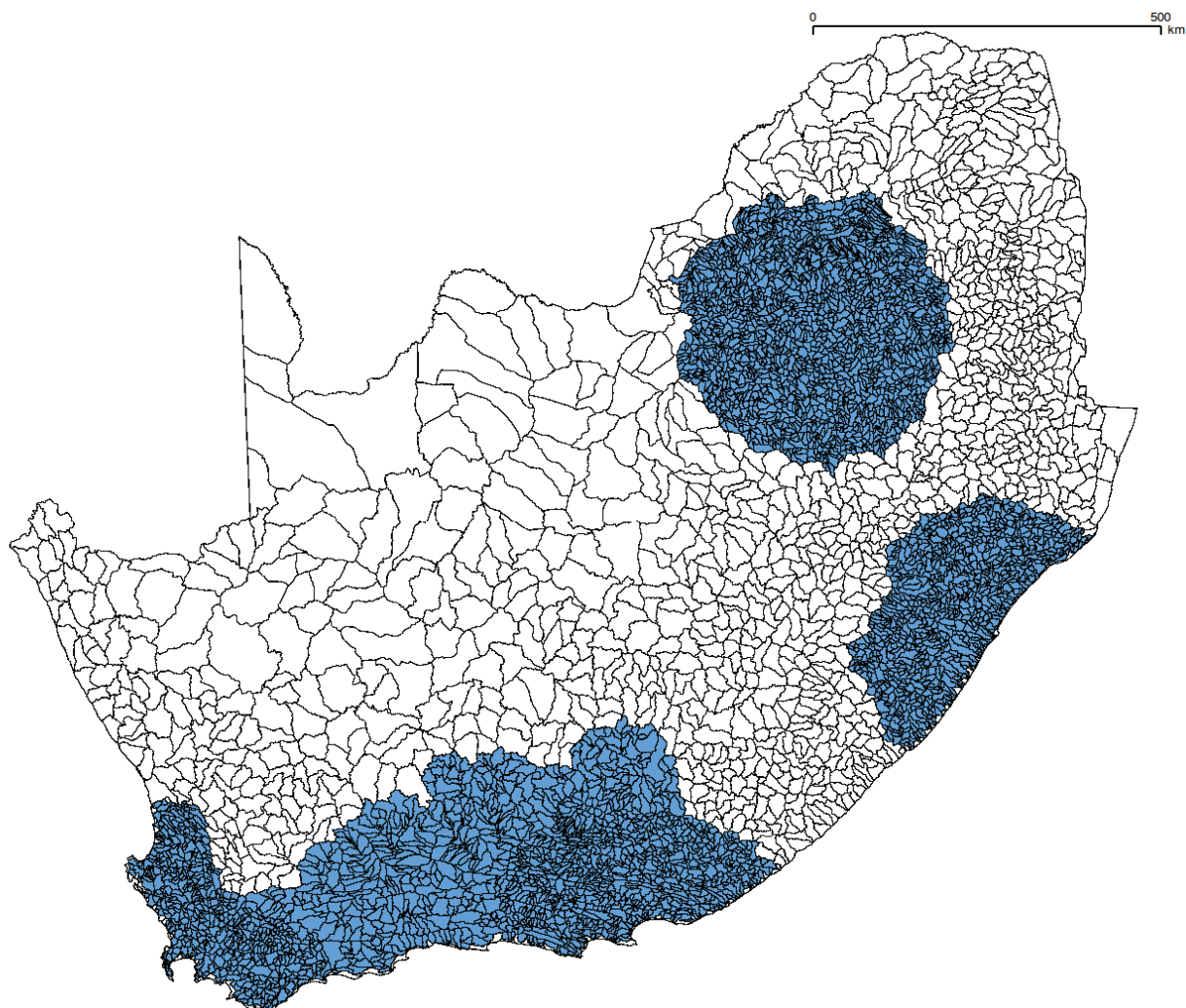


Fig 5.12: Overview of the SAFFG catchments in South Africa, Lesotho and Swaziland. The highly divided SAFFG catchments are the blue coloured polygons, shown overlaid on the WR2005 quaternary catchments. They are clustered within the areas of coverage of the Irene, Durban, Port Elizabeth and Cape Town radars. There are also SAFFG catchments in the fold mountain region of the Western Cape – these are larger in size because the only available rainfall input there is satellite-, not radar- based. [The horizontal bar denotes 500 km.]

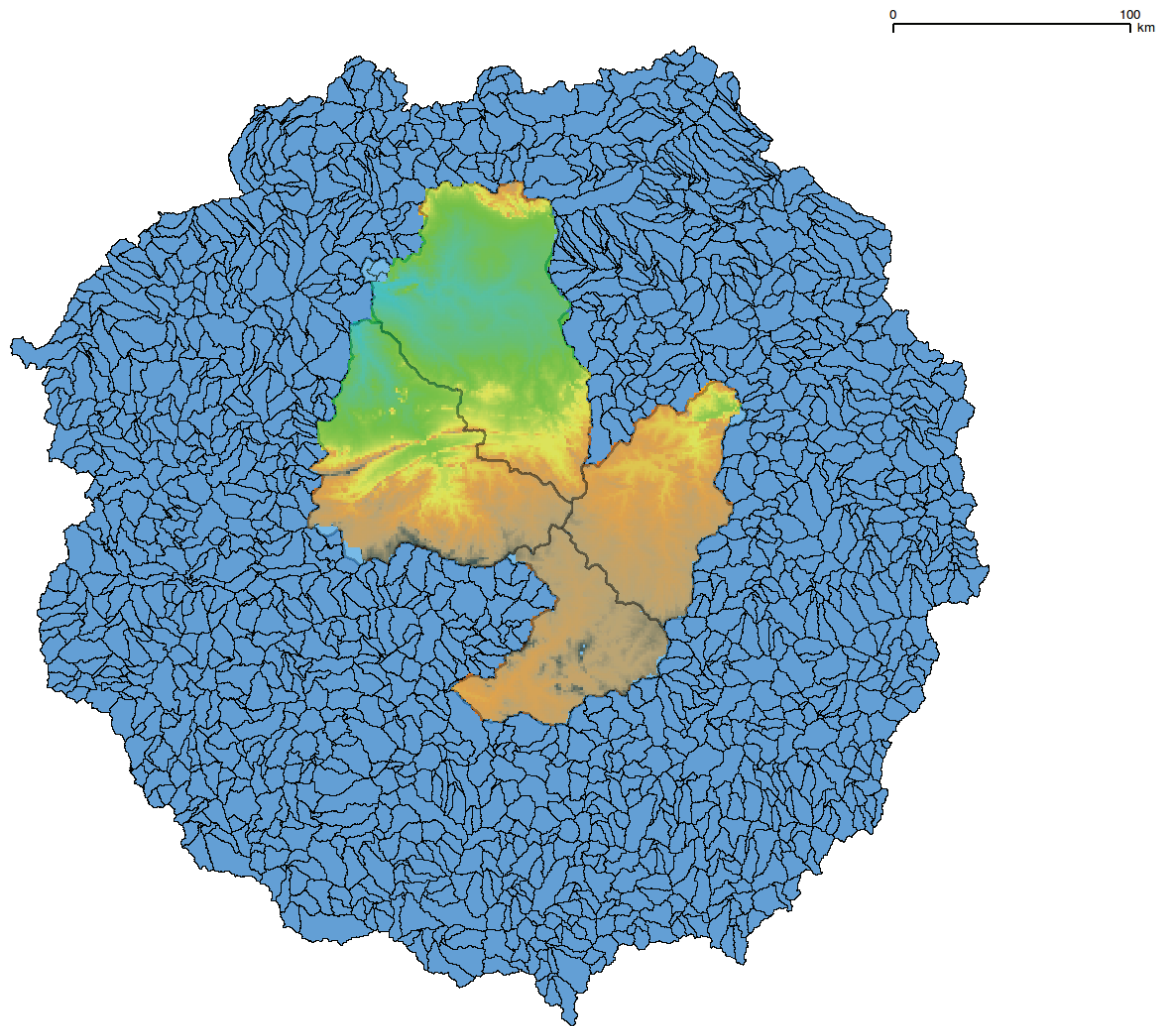


Fig 5.13: A more detailed view of the SAFFG catchments within range of the Irene radar, showing the four WR2005 tertiary level catchments we chose for the modelling process. The elevation is shown by the green (lower) through dark brown (higher) colour scale. Their labelling and areas appear in the Table below. [The horizontal bar denotes 100 km.]

Tertiary catchment	Area (km²)
A23 (North)	6282
A21 (West)	7505
B20 (East)	4360
C21 (South)	3485

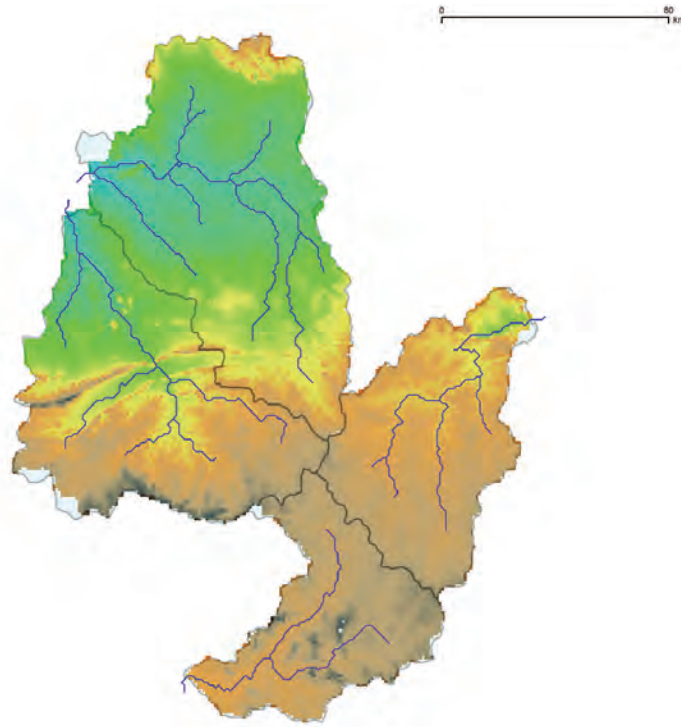


Fig 5.14: A more detailed view of the four extracted catchments coloured in Figure 5.13 and their stream networks, obtained via automated GIS analysis of the DEM, as described in Figures 5.4 and 5.5 and supporting text. [The horizontal bar denotes 80 km.]

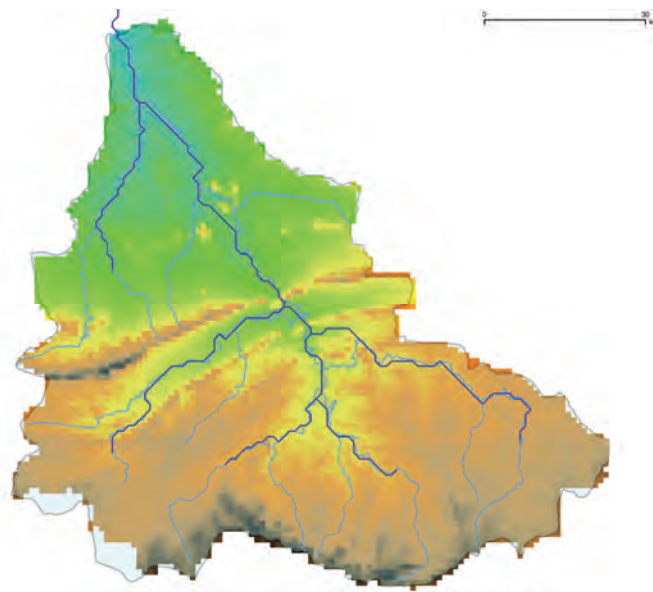


Fig 5.15: A comparison of the stream network (i) automatically extracted by the GIS (dark blue) and (ii) the WR2005 stream network (light blue). This catchment is A21, the westernmost of the 4 subcatchments delineated in Figures 5.13 and 5.14. [The horizontal bar denotes 30 km.]

Several differences are visible between the stream networks in Figure 5.15, but these are currently unavoidable, because the 1 km² resolution DEM we used here cannot properly resolve all drainage paths. We are investigating the efficacy of modelling at higher resolution to resolve these differences.

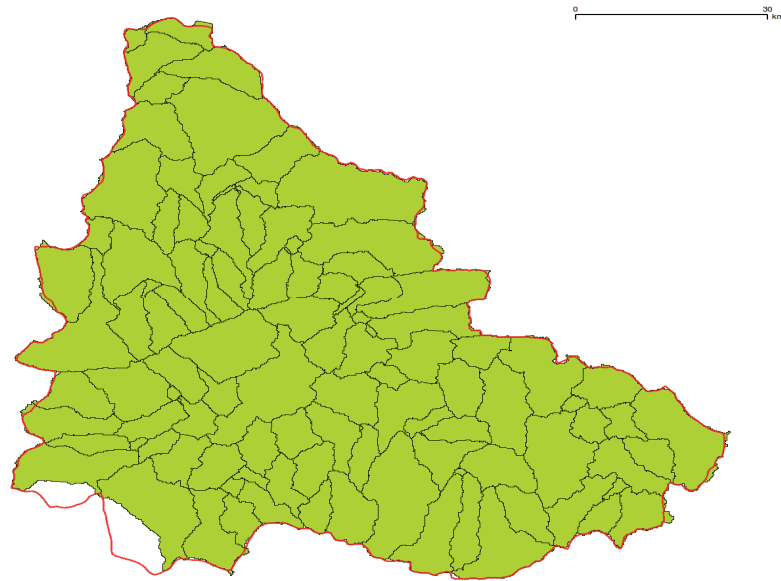


Fig 5.16: Detail of the SAFFG catchments (green) that have the majority of their area falling within the boundary of a WR2005 Tertiary catchment (red outline of Western A21), shown in Figure 5.15. [The horizontal bar denotes 30 km.]

We modelled the tertiary level catchment (shown green in Figure 5.16) using PyTOPKAPI and then computed the average Soil Moisture for each SAFFG catchment from these simulation results. The example of a snapshot of SSI over catchment C21 is shown in Figure 5.17.

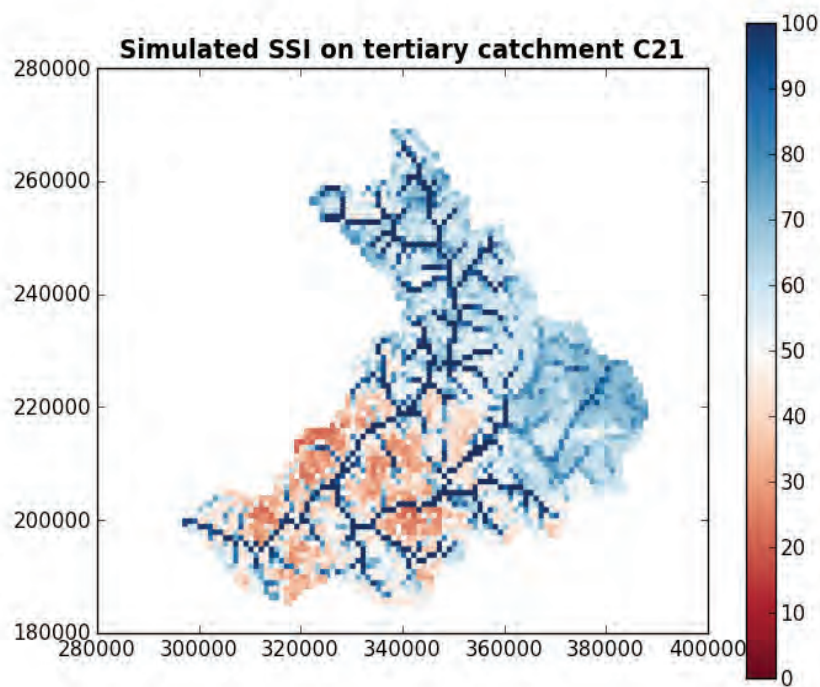


Fig 5.17: SSI modelled by PyTOPKAPI at 1 km^2 resolution for the Southern C21 tertiary catchment shown in Figure 5.13. This is a snapshot of one of the 3 hour time-steps during the 2 year simulation reported here. [The horizontal and vertical scales are in metres.]

The sequence of Figures over the next 3 pages is produced by spatially averaging the SSI over each SAFFG catchment, based on a 3-hour simulation over a period of 2 years. They show the status of SSI averages at the start (00.00 h) on different days. Figure 18 was obtained by spatially averaging Figure 5.17; the others are self-explanatory, assisted by their captions.

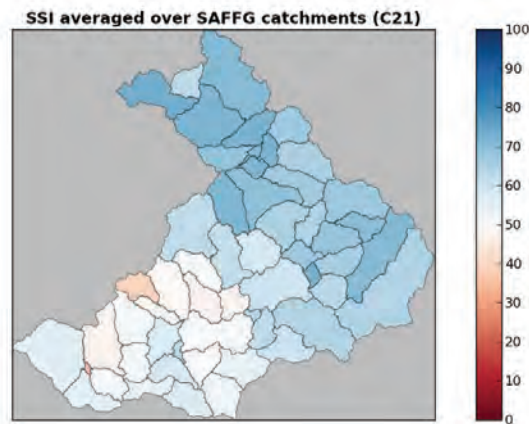


Fig 5.18: The result of averaging SSI over each of the (approximately 60) SAFFG catchments, which fall inside the tertiary catchment C21. This map is computed using the fine-scale modelled results shown in Fig 5.17.

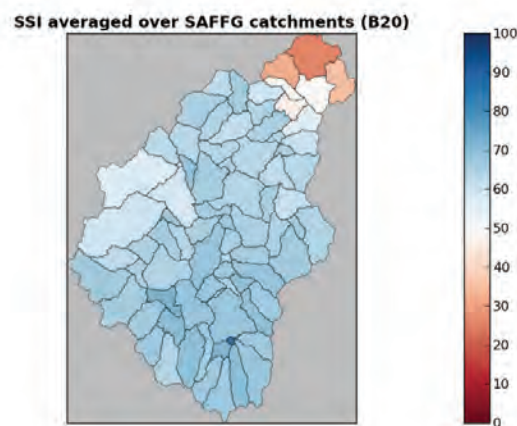


Fig 5.19: As in Figure 5.18, but showing results for the Eastern tertiary catchment B20 (containing approximately 80 SAFFG catchments).

In Figure 5.20, the change in SSI state with time is shown during a wetting-up period. The top-left panel shows the SSI snapshot on day 0, the top-right day 1, the bottom left day 7 (one week later) and the bottom right day 30 (approx. one month later).

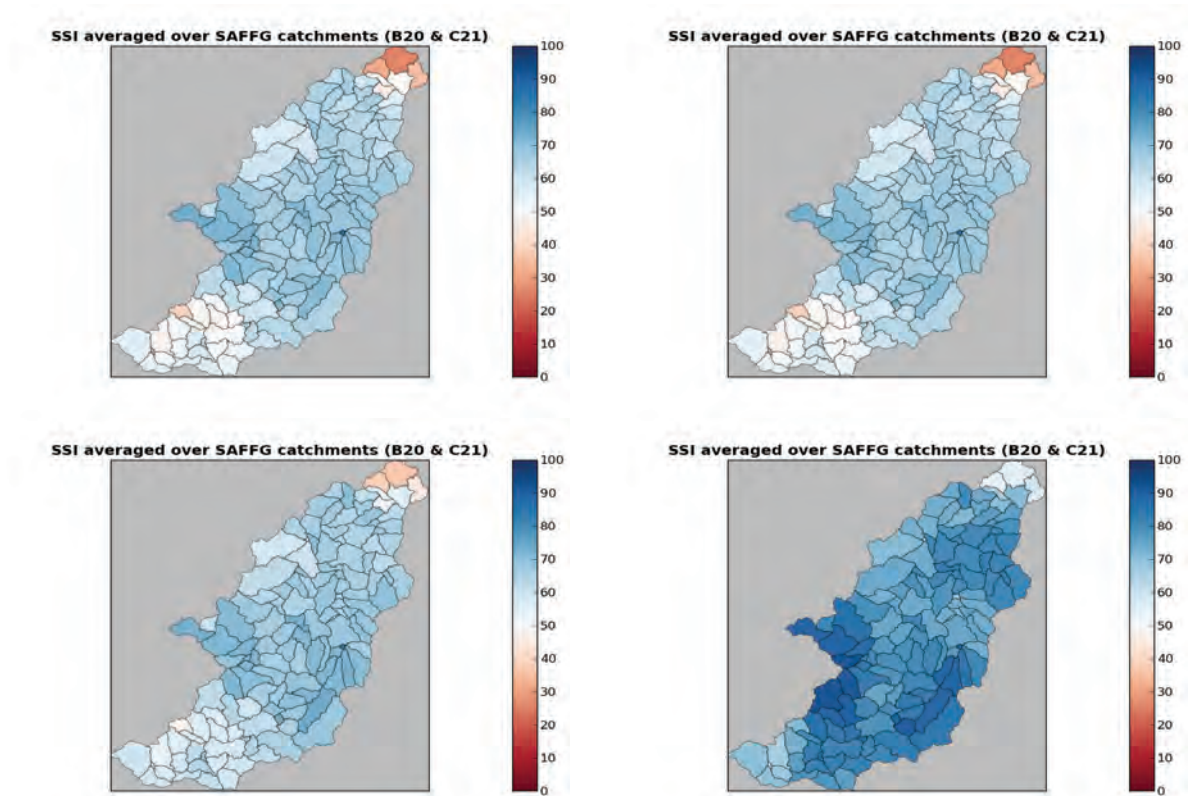


Fig 5.20: SSI is typically a slowly varying quantity. This figure shows the SSI averaged over each of the SAFFG catchments, which fall within tertiary catchments C21 (Southern parts) and B20 (Eastern parts).

In Figure 5.21 the change in SSI state with time, on tertiary catchments B20 and C21, is shown over a calendar year (2010) at monthly intervals. Starting relatively wet in January, peaking in February, the catchment slowly dries out until the beginning of October, then gradually wets up again towards the end of the year. This slow change in SSI response was also a feature of the Liebenbergsvlei, as reported in Pegram et al. (2010) and Vischel et al. (2008a & b).

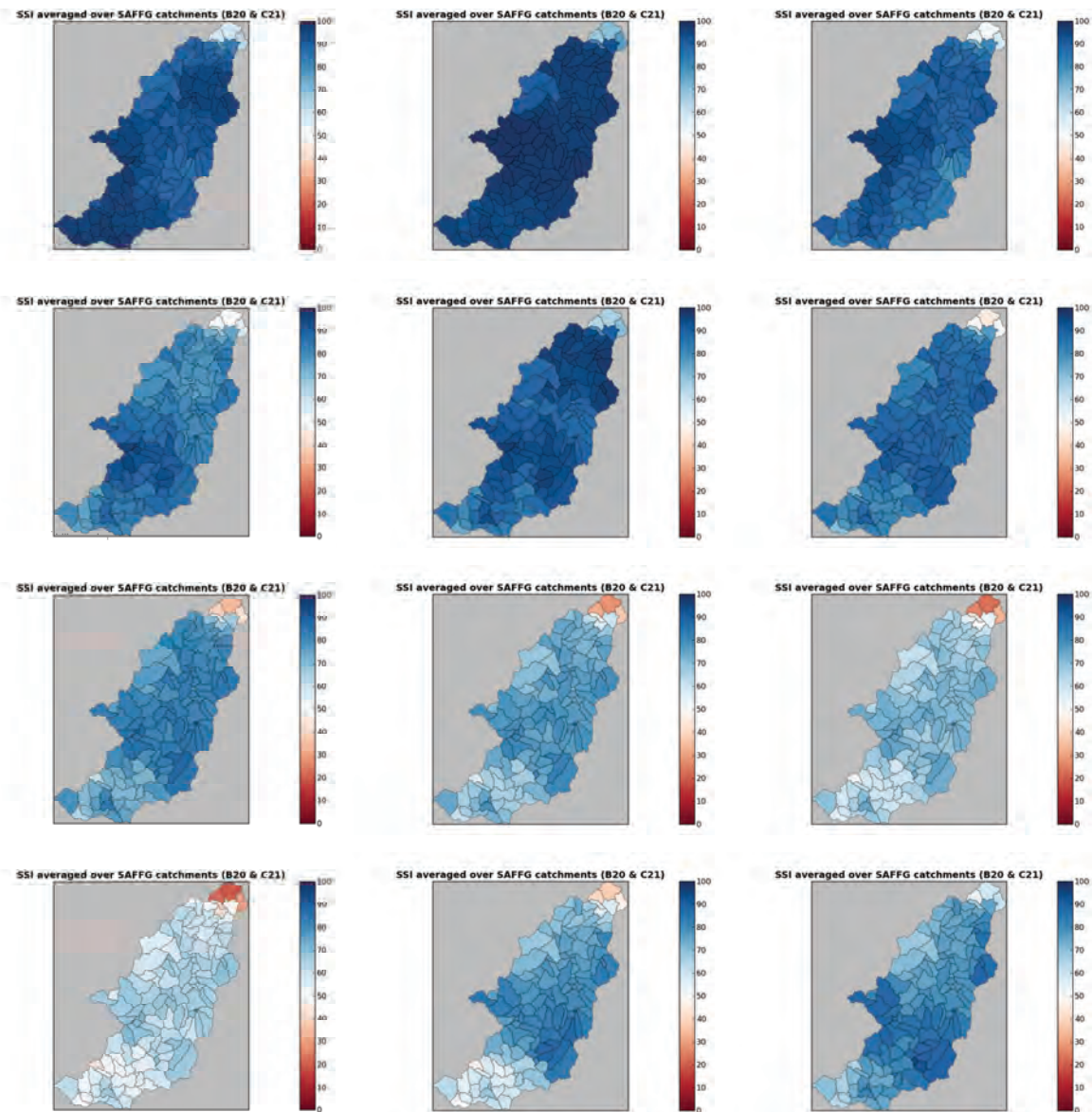


Fig 5.21: Monthly evolution of SSI averaged over (approximately 140) SAFFG catchments lying within tertiary catchments B20 and C21. Each panel is a snapshot of conditions at 00:00 on the first day of the month during 2010. The sequence starts with 1st January 2010 at the top left panel and proceeds in month order from left to right to end with 1st December 2010 in the bottom right panel.

5.3 Summary

The primary goal of this chapter was to report on the work done to model country-wide Soil Saturation Index (SSI) at isolated locations, by setting up hydrologically consistent PyTOPKAPI models at the scale of the SAWS SAFFG catchments. This approach allows detailed modelling of interconnected and contiguous 1 km² cells at fine scale, as an alternative to modelling isolated 1 km² cells at a grid spacing of approximately 12x12 km (as done for the country-wide Soil Moisture assessments). The eventual aim is to assess the role of lateral transfers of water between cells in the modelled SSI.

The work done to achieve this goal consisted of two components 1) develop tools to (partially) automate the process of setting up a PyTOPKAPI model in new catchments, so that fine scale modelling of many SAFFG catchments becomes feasible, and 2) to test these tools by using them to do detailed modelling on a selection of SAFFG catchments.

The success of the adaptation has been clearly demonstrated in this chapter, giving us a solid software platform to work on, and assisting new model users to set-up PyTOPKAPI in their own catchments.

6: Back calculate SSI estimates over the Period of Available Data

Preamble

The original intention of this chapter is to regenerate Soil Saturation Index (SSI) simulation results using the updated static parameter maps and forcing variables, from earlier work done in a desktop study, as a once off effort. However, at the time the project proposal was written we did not know that this would actually become a regular monthly exercise. At the outset of this project we were invited to contribute monthly Soil Moisture assessments to the Umlindi newsletter published by the Agricultural Research Council's Institute for Soil, Climate and Water (ARC-ISCW). The first assessment was for February 2010 and these contributions have been on-going with the most recent for July 2012. All of these reports are archived on our website at <http://sahg.ukzn.ac.za/reports/>. Throughout the course of the project we have applied continual improvements to the SSI assessments by including the innovations developed as they became mature. We have now reached the stage where the complex process of updating the SSI simulation and posting the latest results for download from our website is effected by issuing a single command. The generation of the SSI assessments has also been greatly streamlined by developing scripts to generate the relevant figures and the report template. This report outlines the work done to achieve all of this in order to meet (and materially exceed) the original expectations.

6.1. Overview of the successive improvements to the SSI simulations

In the work reported in chapter 2 we updated the static parameter maps that are used by the modelling procedure. Our original data-set was based on the SIRI (1987) soil maps, which contained large data gaps over the parts of the Eastern Cape – the former Transkei and Ciskei. We updated these parameter maps based on the information provided by Schulze et al. (2008), to remove these gaps and use nationally accepted standards for the soil properties that drive the PyTOPKAPI model in HYLARSMET mode. Figure 6.1 shows the SSI map averaged for January 2010 computed using the two different static parameter sets with the same forcing applied.

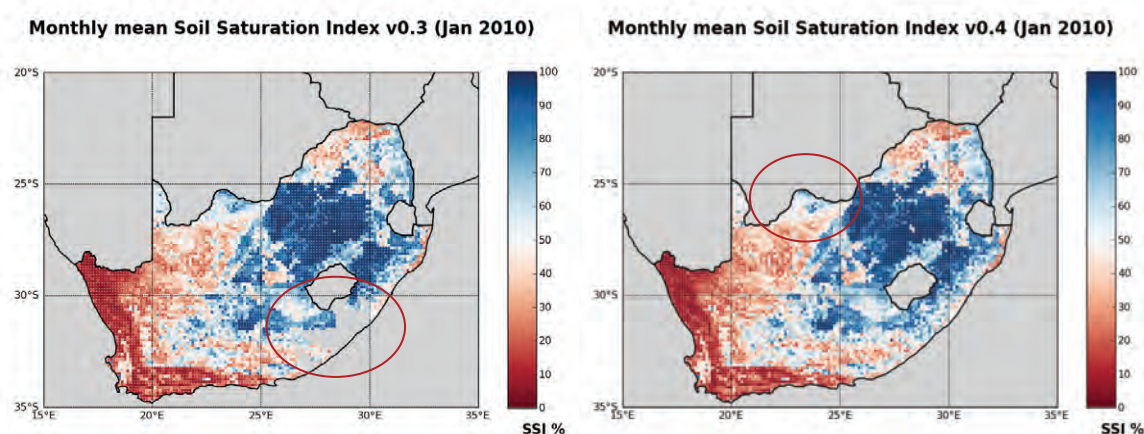


Figure 6.1: Comparison between the SSI results obtained for two different base parameter sets. The SSI shown here is the monthly average of 3 hourly simulated SSI for January 2010. The new calculation is on the right. The new algorithm version used is v0.4 (using the updated parameter maps). There are no substantial differences between the SSI simulations of algorithm v0.4 and those using algorithm v0.3, except that SSI is now a little lower in the Northern Cape due to the deeper soil estimates. The main difference is that the missing soil properties over the Eastern Cape have been infilled by reanalysis using the data from Schulze et al. (2008). These issues are shown ringed by ellipses in the two images.

In Chapter 3 we reported on the structural changes made to the PyTOPKAPI model in order to include an infiltration module based on the robust and well known Green-Ampt (1911) method. In that chapter we showed that infiltration does not have a large effect on SSI at 3 hourly time-steps for most of the country, but that there are a few places with extremely clay-rich soils, where including the infiltration process has a pronounced effect on the SSI. We also showed that the effects of infiltration are most important when the model is run at shorter time-steps (i.e. for flood forecasting applications). The HYLARSMET SSI modelling system now uses a version of the PyTOPKAPI model which includes Green-Ampt infiltration as one of the modelled processes.

In Chapter 4 we investigated the sensitivity of the HYLARSMET SSI modelling process to changes in the forcing variables and static parameter data-sets. The main result we reported there is that the modelled SSI is most sensitive to rainfall forcing (Sinclair and Pegram, 2013). The 3 hourly TRMM 3B42RT rainfall product (Huffman et al., 2007) has undergone several algorithm evolutions during the period of this project and we have updated the code and data gathering processes to ensure that we use the most mature rainfall product available for each modelling time-step. The latest update of the TRMM algorithms by NASA to version 7 will shortly include back processing of the entire rainfall archive to 2000 (Huffman, 2012, personal communication), this processing has already begun (Stocker, 2012, personal communication) and we will soon be able to use a consistent rainfall product for the entire period of our simulations.

Figure 6.2 is an updated version of our flow diagram, reflecting the changes incorporated and described in the above paragraphs.

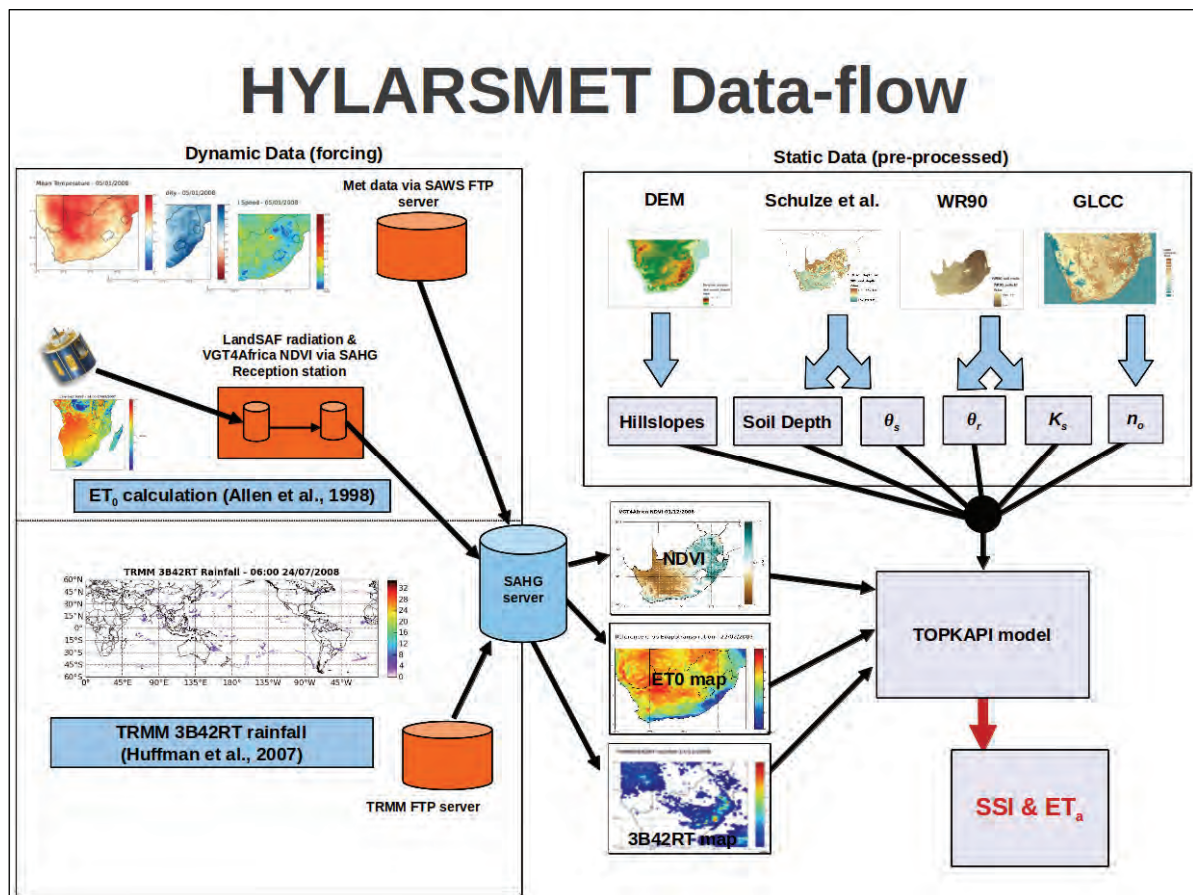


Figure 6.2: Updated data-flow diagram showing sources of the dynamic data (left panel) and static data (upper right panel) to produce the main information streams: Soil Saturation Index and Actual Evapotranspiration (bottom right).

6.2. Description of SSI back-calculation process

This section describes the complicated process of updating the SSI simulations and data-sets in some detail.

6.2.1 Data collection and archiving

Collecting and archiving the forcing data components, and their input data streams, is an on-going process, with the source data-sets coming from several different locations and via several different delivery mechanisms. All the software for collecting and archiving these data-sets was coded in house.

The first forcing variable collected is rainfall. TRMM rainfall data is automatically downloaded from the NASA FTP server to the SAHG server at the University of KZN. The download script is set to query the NASA server at 20 minute intervals and download any new data files that are found there and add them to the archive on the server. Nightly backups of this data are made, now automatically, to an independent hard drive on the local server, to guard against disk failure. The data are provided in a custom binary file format, and we have developed an open source Python package to read the data into memory for further processing. The PyTRMM package can be obtained from <http://github.com/sahg/pytrmm>

The second model forcing variable is reference crop evapotranspiration (ET_0), which is calculated using the FAO56 method (Allen et al., 1998). The variables that are used in this computation must be collected and archived. The first of these is an estimate of solar radiation flux. We obtain this stream from the Eumetsat LandSAF Down-welling Surface Short-wave radiation Flux (DSSF) product generated from Meteosat data. This arrives via a EUMETCast C-band satellite feed at 30 minute intervals in near-real time and is collected in a rolling 10 day buffer on our EUMETCast receiving station. A process running on the receiving station once every hour was designed to copy all new data across to the SAHG server where it is archived and, in addition, a backup copy is made once daily to a separate hard disk on the server. The other variable components that go into computing ET_0 are obtained from the SA Weather Service's Unified Model runs. At our request, SAWS runs a post-processing step after their 00:00UTC model run that clips out a portion of their model field for three variables and copies these data files to their FTP server. The three variables (temperature, relative humidity and wind) are collected once daily to the SAHG server and, as with the other data-sets collected nightly, backups are made to a separate disk.

The final forcing variable we use is NDVI in order to adapt the ET_0 and compute ET_a in a coupled calculation with the PyTOPKAPI hydrological model. The NDVI we use is the 1km VGT4Africa product which is updated at ten day intervals. The data files arrive via our EUMETCast system and are copied onto the server where they are archived for use by the HYLARSMET SSI modelling system.

6.2.2 Update reference crop evapotranspiration

The ET_0 estimates are calculated from the relative humidity, temperature and wind fields as well as the DSSF radiation product at hourly time-steps using the algorithms described in FAO56 (Allen et al., 1998). The calculations are automatically updated once daily. The previous days ET_0 estimate is calculated on a 0.11 degree regular latitude-longitude grid and added to the archive. Each month's results are stored in separate HDF5 files that are backed up to a separate disk each night.

6.2.3 Update SSI simulations

The previous discussion described how the forcing data is gathered, prepared and archived using automated processes. Once the forcing variable archives are in place it is possible to run an SSI simulation that ends with the date of the most recently available forcing data. Prior to the completion of the work reported here, it was a time-wasting necessity to re-run a complete simulation each time an update was required (i.e. every month for our contribution to the Umlindi newsletter). Since a full simulation run takes close to two days on the hardware we have available, it became very tedious to do regular back calculations, which gave us the drive to make this good.

Enhancements have now been made to the code that runs the HYLARSMET system to enable a new simulation to do a 'warm' start up, using the model state from the most recent time-step and continue the calculations from there. The modelling system must also be executed in a virtual environment that ensures that the correct software versions and tool chain are used each time, making the process reproducible. A wrapper script has been developed that sets up the virtual environment before executing the simulation updates. Running updates now takes less than an hour (typically updates are manually initiated once a month, but are going to be completely automated in the future).

6.2.4 Update available data on the website

All of our generated SSI and ET_0 results are available for viewing and download on our website (http://sahg.ukzn.ac.za/soil_moisture/). The data available on the site was previously updated manually at infrequent intervals, because it involved a large investment of manual work to convert the simulation results from their original format and place them on the SAHG web and FTP servers. The work reported here consolidates that effort into a set of scripted and easily repeatable actions. The result is that we've turned a large suite of research-oriented code into efficient, easy-to-use, sharable code. The flow chart depicted in Figure 6.3 shows how an outside user may gain access to the archives of the images and the data which created them.

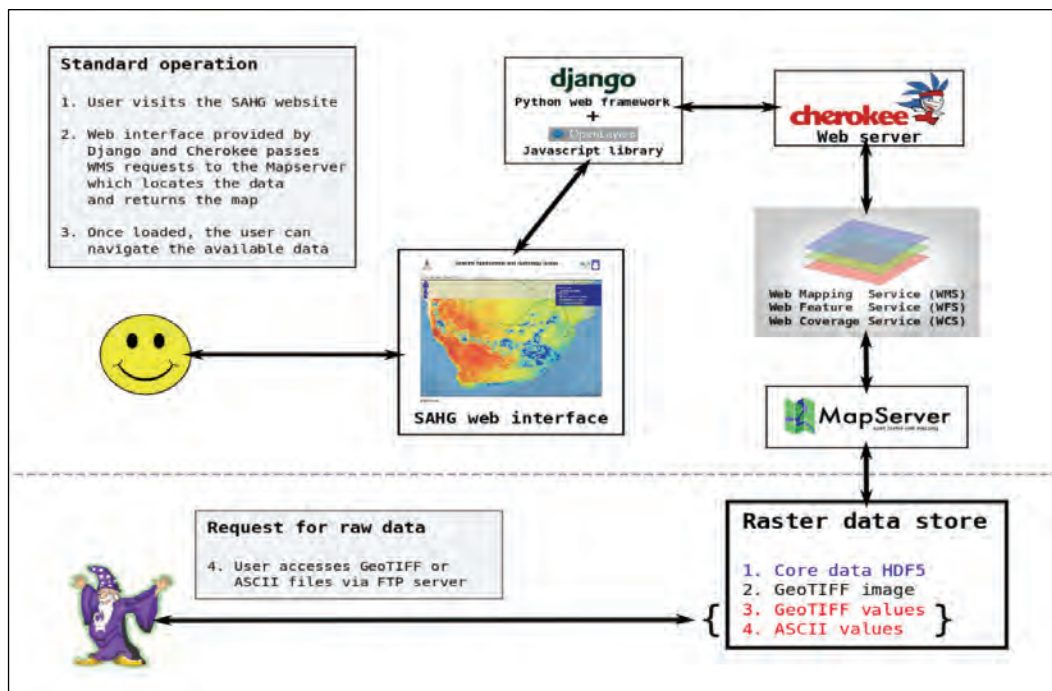


Figure 6.3: The flow chart indicating how a casual user (upper panel) or a more serious user (lower panel) will gain access to the images and/or the data used to create them, stored on the SAHG servers.

The most recent set of ET_0 and SSI results are copied to the workstation used to do the data processing (only differences are copied to avoid taking too long for the data transfer). A conversion is then run to generate SSI interpolated onto a regular grid matching the SAWS Unified Model grid. As shown in Figure 6.3, both SSI and ET_0 results are produced in three different gridded geo-referenced formats for display and download from the SAHG webserver. The first is an image based GeoTIFF format used by the map-server to display sample data on the website's data viewer. Second is a floating point GeoTIFF format suitable for download and display in geo-spatially aware software (e.g. GRASS GIS, ArcGIS etc.). An ASCII grid format is also produced to allow easy access to (i) programmers without experience of using GeoTIFF libraries, and (ii) users of ArcGIS versions earlier than version 10.0 (or indeed, other software). Once the gridded data files have been prepared on the local workstation, they must be copied to the appropriate places on the SAHG web and FTP servers and the map-server configuration file must be updated in place to allow the web data viewer to access the new data. All of this used to be done by hand, on request, but is now automatic (and carefully scripted) to ensure the process is consistent. This large investment in customised coding of software means that unnecessary copying and data conversion will be avoided in future; in addition it will reduce the time taken to complete the process.

6.3. Results

Because the tangible results of the work reported here are primarily the on-going Soil Moisture assessments in the Umlindi newsletter and the scripts and software upgrades that we have made to allow simple and frequent back-calculation of the SSI estimates, we present some results of the modelling procedure in this section. Figures 6.4 to 6.10 that follow are fully explained by their captions, and show that SSI is more slowly varying than the ET_0 estimates that we calculate during the modelling procedure (initially described in Pegram et al., 2010). The time intervals displayed increase from the basic calculation interval of 3 hours, dictated by the TRMM rainfall product frequency, through daily and monthly to annual.

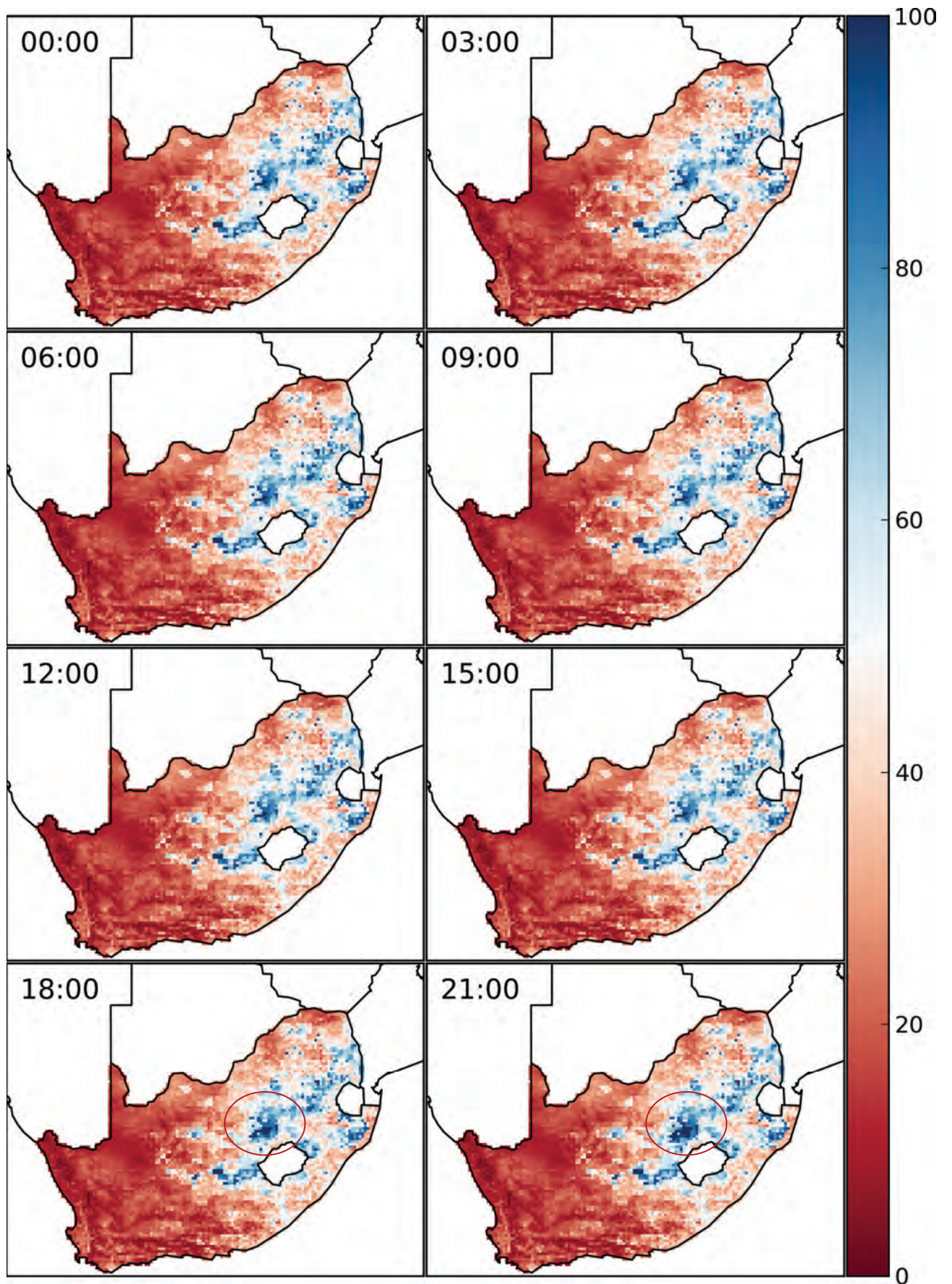


Figure 6.4: Evolution of SSI at 3 hourly intervals during a single day (2008/11/10). Note very little change from one panel to the next, apart from a slight increase in SSI over the central Free State, shown ringed in the last two panels, following heavy afternoon thunder showers.

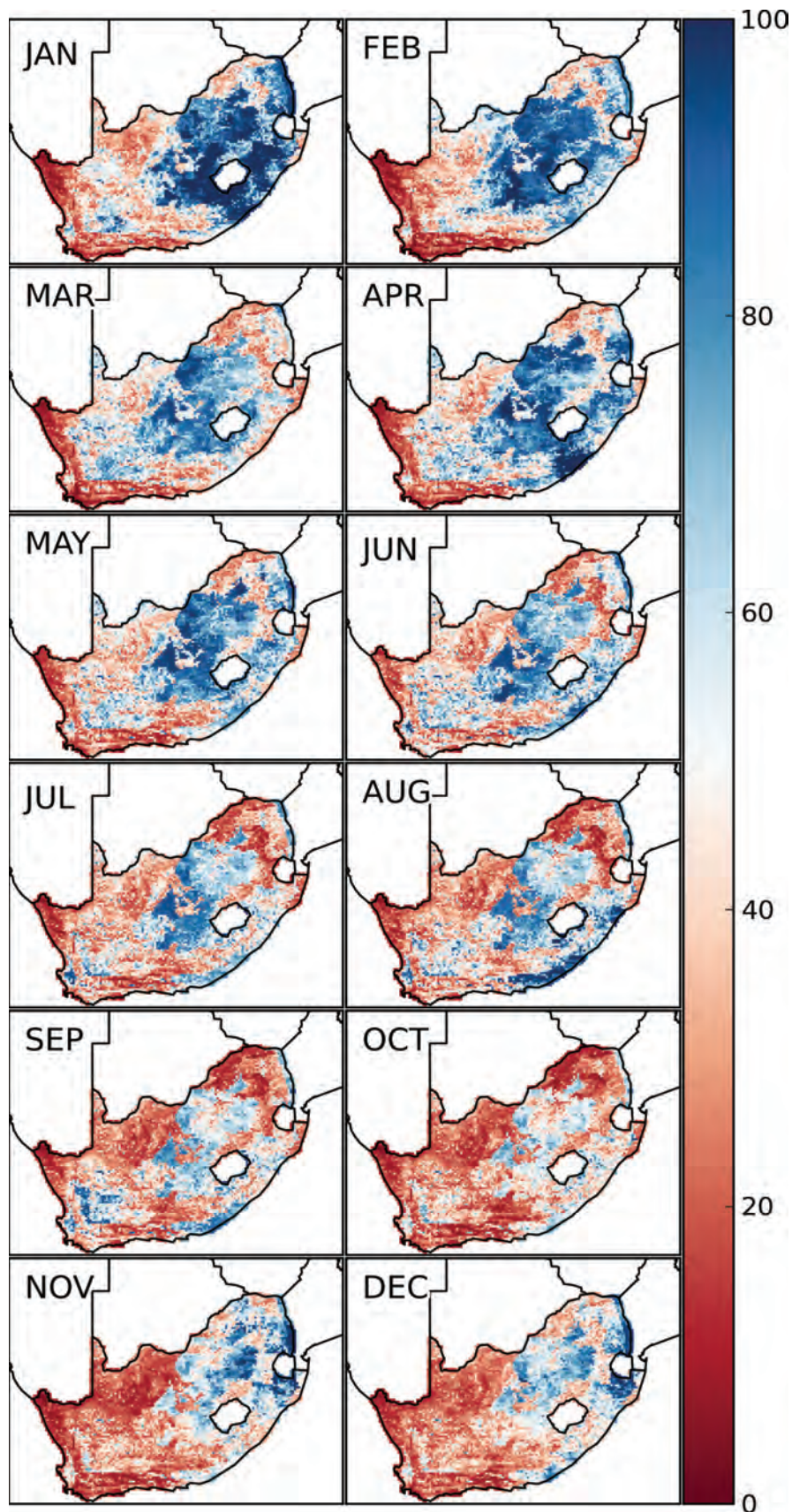


Figure 6.5: Snapshots of SSI at the start of the first day of each month during 2011. Note the significant seasonal changes after the rains end in the interior, but at a very slow rate with a half- life of the order of 4 to 6 months. It is clear that the Eastern Interior starts wetting up after the October rains.

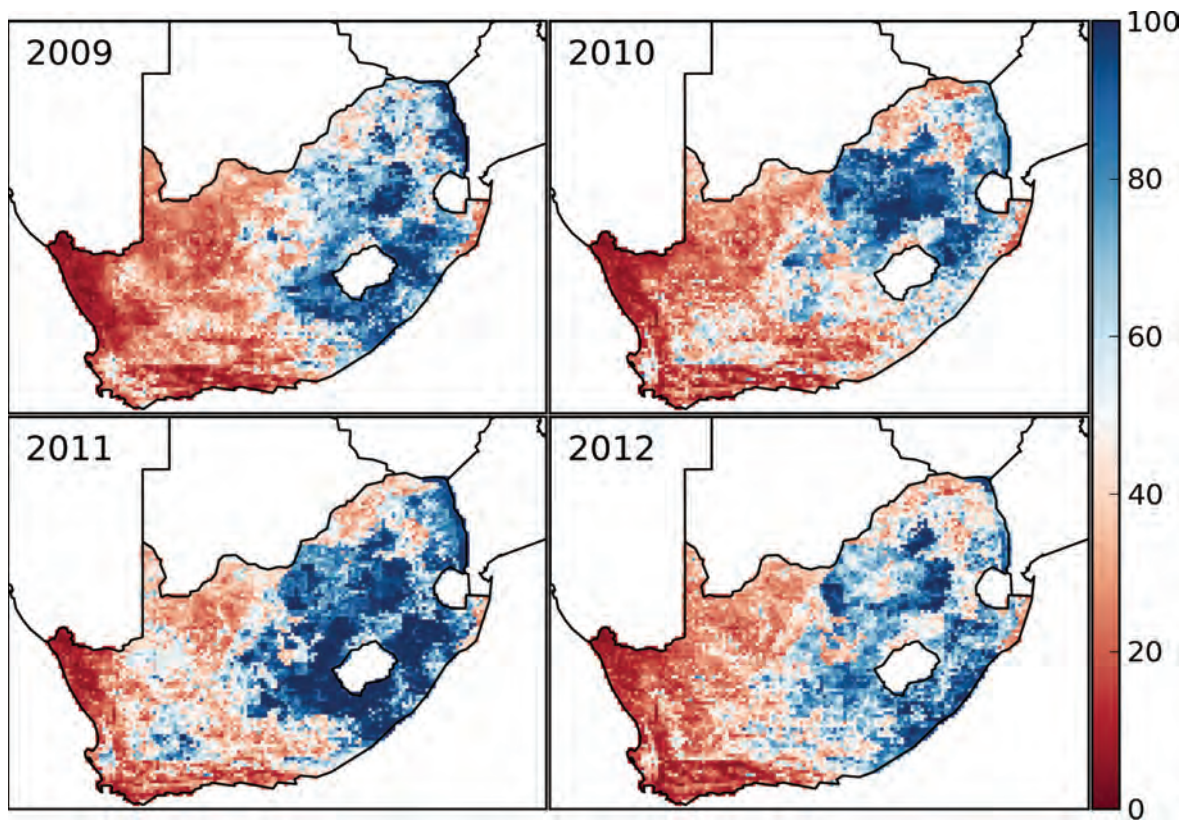


Figure 6.6: Annual variability in SSI for snapshots on 1st January each year.

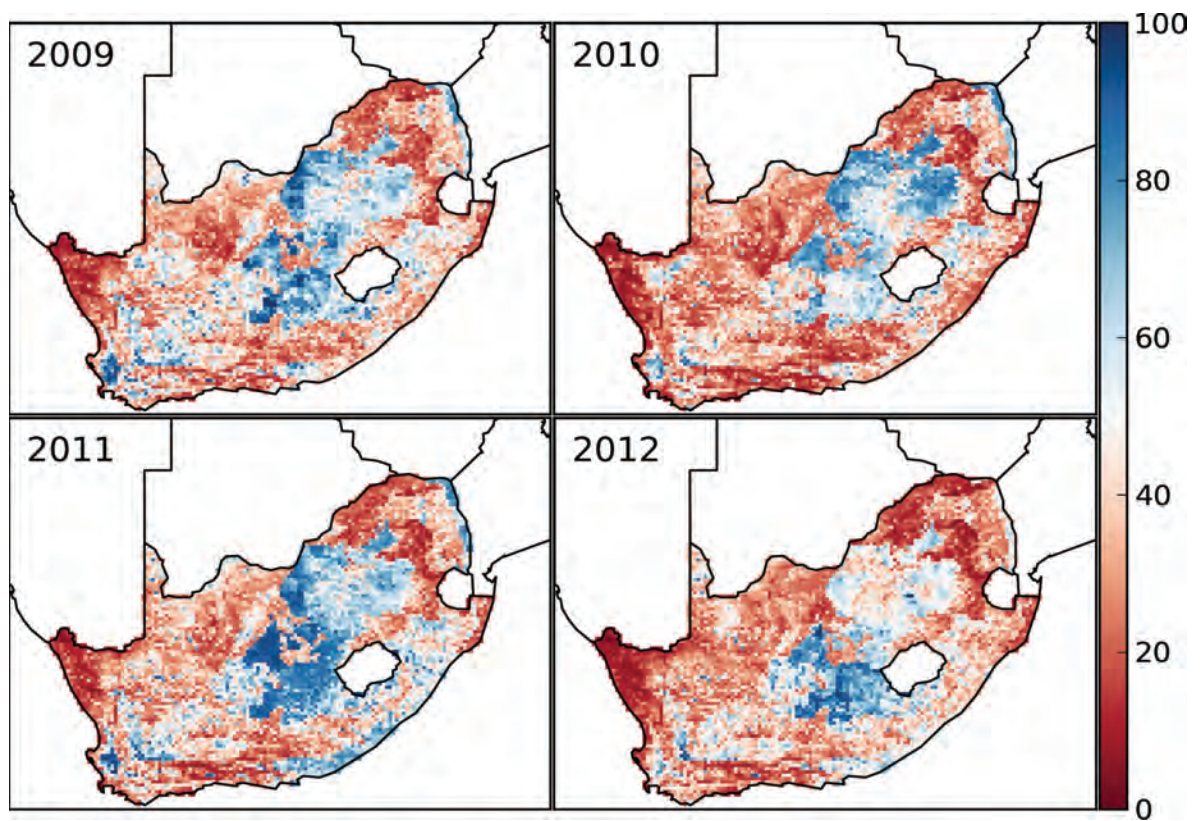


Figure 6.7: Annual variability in SSI for snapshots on 1st July each year, on average much drier than January in Figure 6.6 above (except in the Western Cape).

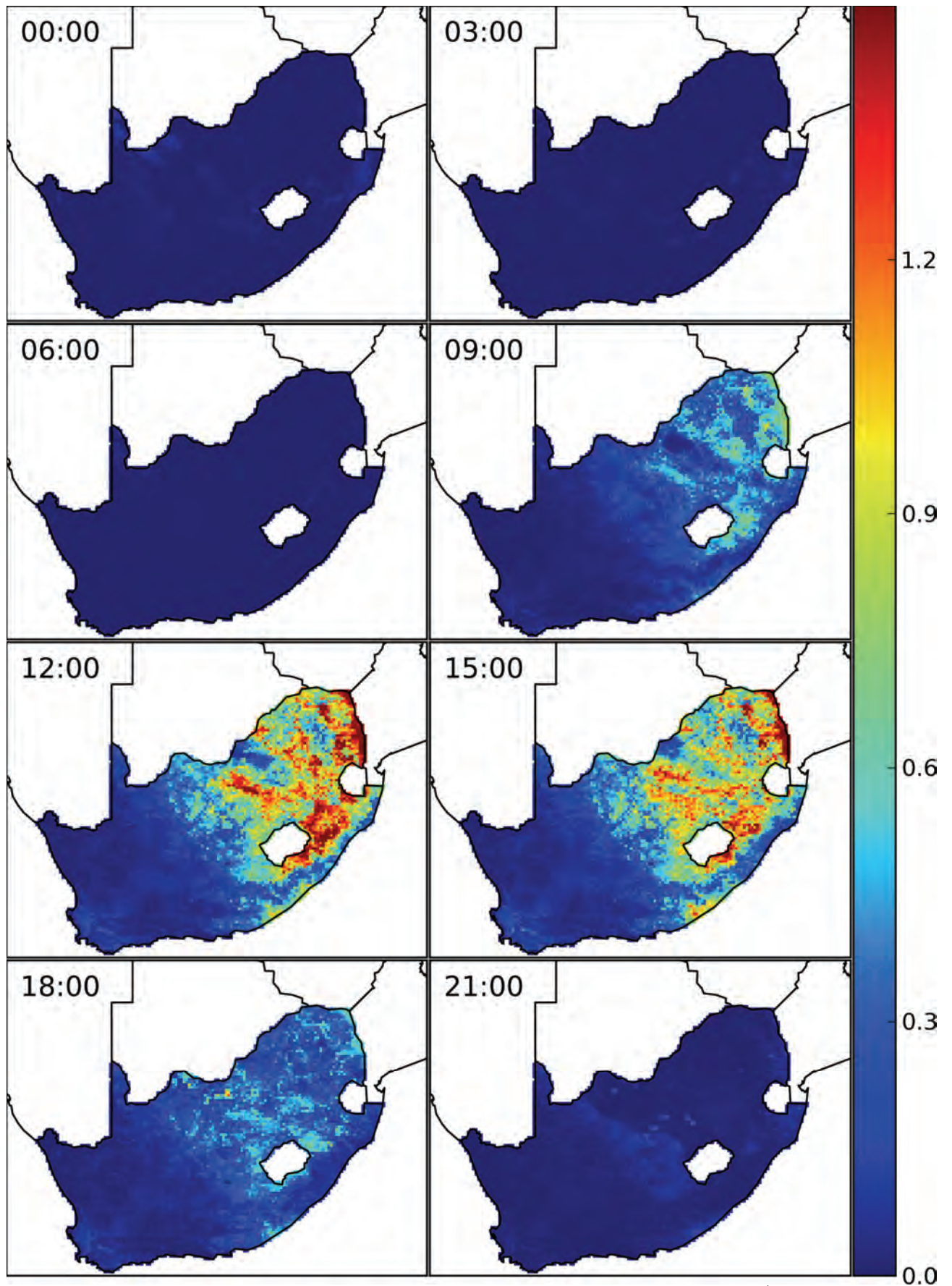


Figure 6.8: Three hourly accumulations of actual evapotranspiration (ET_a) for 2nd February 2009, in millimetres matching the colour code to the right of the panel. Note the high variability in space and time, with negligible or very low values during the night time.

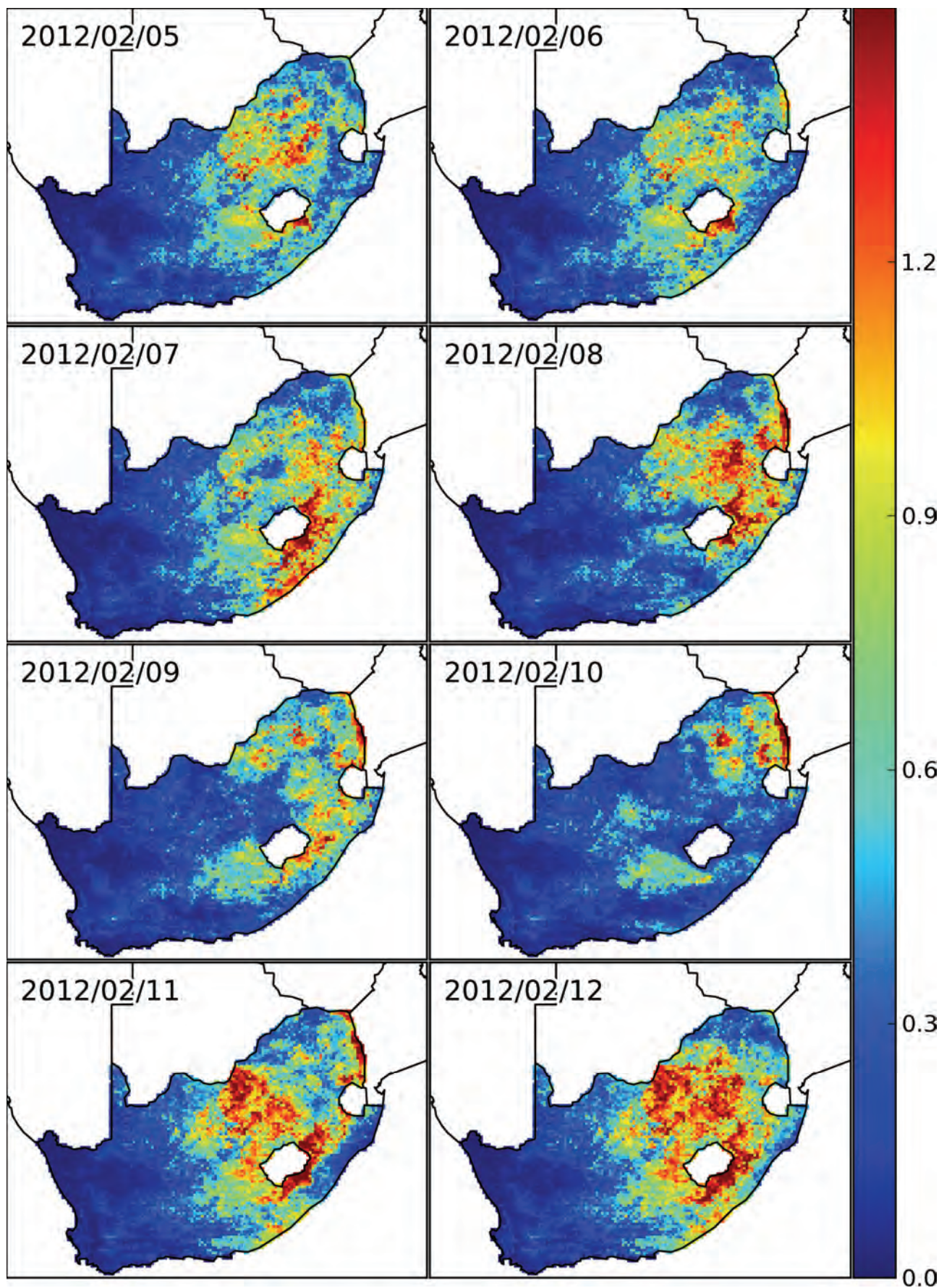


Figure 6.9: Daily evolution of 3 hour ET_a accumulations (mm), (like those shown in Figure 6.8, so that the scales are the same in each figure), but for the 3 hour period ending at 12:00 on each of the 8 days.

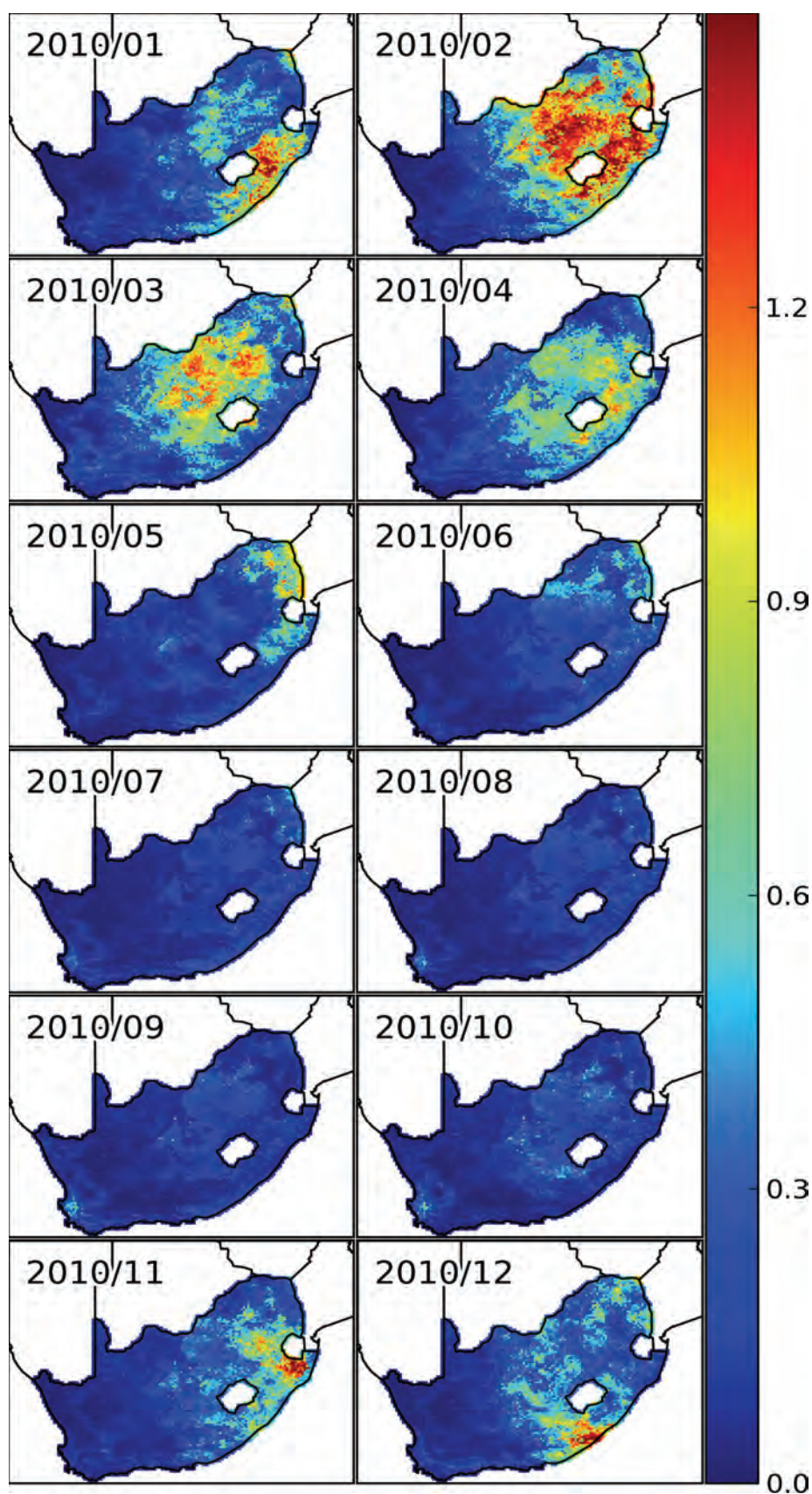


Figure 6.10: A 1-year sequence of snapshots of 3 hour ET_a accumulations, at a monthly spacing (3 hour period ending 12:00 on the 3rd of each month). The strong seasonal change over the year is evident in the areas usually experiencing relatively high rainfall – Winter rainfall in the Western Cape and Summer rainfall in the Eastern part of the country. Unlike ET_o , ET_a does not occur if the soil is dry.

6.4. Summary

The original intention of work reported here was to regenerate Soil Saturation Index (SSI) simulation results, using the updated static parameter maps and forcing variables, in a desktop study as a once-off effort. We have exceeded this original intention by first doing the back-calculations on a monthly basis for the ARC-ISCW Umlindi newsletter in an ad-hoc way by hand, and then developing a tool-chain to automate this process in an easy and repeatable manner. Throughout the time that we have been producing monthly Soil Moisture assessments we have been archiving the source data-sets and building on the work done earlier in the project, in order to improve the HYLARSMET SSI modelling procedure. The primary evidence of this work is that all of the historical and most recent results are available on our website at http://sahg.ukzn.ac.za/soil_moisture/ and can be downloaded by interested users. We will continue to update the products for the duration of this project and excitingly have negotiated funding with the World Meteorological Organization to assist us in transferring the full HYLARSMET system to the ARC-ISCW for operational use going forward.

An example of the Umlindi offering we produce every month appears under Section 8.4.1 headed "Continued contributions to ARC's monthly Umlindi report."

7. Validate SSI estimates against other Models

Preamble

It was suggested, at the last meeting of the precursor project K5/1683, by Prof Bruce Hewitson of CSAG UCT, that the HYLARSMET product has useful potential for assimilating Land Surface Model variables into Regional Climate Models. Therefore, for the purpose of moving beyond our borders to include countries preferably as far as the equator (but at least SADC), we need to compare the results in RSA with those driven by a global data set.

The outcome of the work reported in this chapter makes this suggestion a reality and puts structure to the 6th Aim of the project as expressed in the contract document, to “Validate the methodology developed and implemented”.

The methodologies against which we make comparisons of the PyTOPKAPI Soil Moisture estimates, derived in this WRC K5/2024 project HYLARSMET, come principally from a two-year European Union FP7 project called GLOWASIS [Global Water Scarcity Information Service] in which we have been partners in our capacity of UKZN researchers, during the period Jan 2011 to Mar 2013. Although the GLOWASIS products are aimed at water scarcity, one of the core drivers of the methodologies created to obtain measures of this signal was remote-sensing of Soil Moisture and global hydrological modelling of catchments and thus soil water behaviour.

Nested in the Global product was a particular emphasis on Africa, hence we were able to make selected model intercomparisons of daily Soil Moisture (SM) estimates over Southern Africa during a two-year overlapping time frame. Although the common period is relatively short, we have been able to show that there is pleasing correspondence between HYLARSMET/PyTOPKAPI on the one hand and the GLOWASIS products on the other, especially the modelled outputs with similar rainfall forcing (i.e. TRMM), also providing broad agreement over the whole set. In addition, it was of value to be able to compare the data sets of fundamental soil properties driving and defining the models. Here again, relatively good agreement at the scale of 0.5° (about 50 km) was realised, suggesting that PyTOPKAPI in HYLARSMET mode may be validly used outside RSA borders, where in many African countries only the FAO soil property data sets are available to us.

7.1 What are the sources of data for the model intercomparisons?

We have compared HYLARSMET’s Soil Saturation Index (SSI) to a variety of modelled and remote-sensing Soil Moisture estimates during the time period 2009/01/01 – 2010/12/31 at daily time-steps. Data-sets for comparison which were obtained via GLOWASIS (in which we were invited African partners as a result of the work we have done using WRC funding) include PCR-GLOBWB model simulation results, with 3 different rainfall forcings applied (TRMM, PERSIANN, ERA interim), TUWIEN relative Surface Soil Moisture (SSM) and NEO’s DRYMON-SSI profile Soil Moisture estimate.

The estimates listed hereafter were at different spatial scales therefore all had to be re-sampled to a common grid for the comparison (0.5° regular Latitude/Longitude grid selected to match the PCR-GLOBWB grid).

PCR-GLOBWB global hydrological model

The PCR-GLOBWB [PCRaster **GLOBAL** **Water** **Bal**ance] model is the hydrological model devised by Van Beek and Bierkens (2008), of the University of Utrecht (some of our GLOWASIS partners), and used in nowcast mode to produce global simulations of hydrological variables (including Soil Moisture). The model is used in seasonal prediction, modelling hydrological effects of climate change, water level modelling for CH₄ emission from wetlands, modelling of terrestrial water storage and modelling global water resources in the context of water stress. A sample of their products is taken from their report and is shown here as our Figure 1.

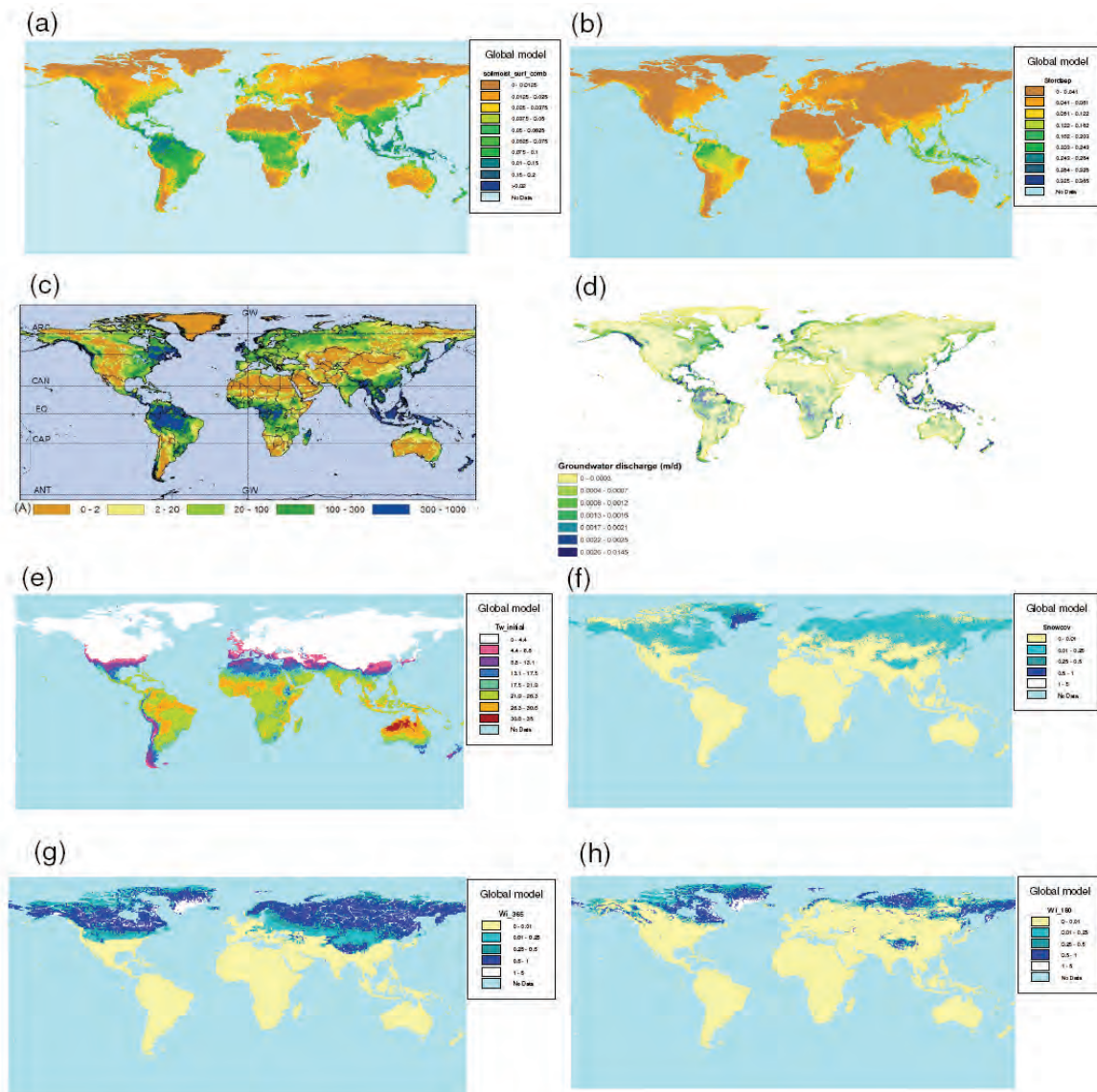


Figure 7.1. Some collected additional (unverified) output from PCR-GLOBWB; (a) average soil moisture content on January 1; (b) average soil groundwater storage on January 1; (c) average groundwater recharge (mm/year); (d) average yearly groundwater discharge to surface water (i.e. baseflow; mm/day); (e) average surface water temperature on January 1 (°C); (f) average snow cover on January 1 (m water equivalent); (g) average ice thickness (m) on January 1; (h) average ice thickness (m) on July 1.

In this chapter our interest is confined to the panel (a) in Figure 7.1 which shows the global average Soil Moisture content on January 1.

PCR-GLOBWB is a large-scale hydrological model intended for global to regional studies and provides a grid-based representation of terrestrial hydrology with a typical spatial resolution of about 50×50 km (currently 0.5° globally) on a daily basis. Similar to other large-scale hydrological models, PCR-GLOBWB is essentially a leaky bucket type of model applied on a cell-by-cell basis (i.e. no lateral connections between cells – an innovation which PyTOPKAPI can bring to large scale hydrological modelling). For each grid cell, PCR-GLOBWB uses process-based equations to compute moisture storage in two vertically stacked soil layers as well as the water exchange between the soil and the atmosphere and the underlying groundwater reservoir. Exchange to the atmosphere comprises precipitation, evapotranspiration and snow accumulation and melt, which are all modified by the presence of the canopy and snow cover. The exchange with the underlying groundwater reservoir comprises deep percolation and capillary rise and vertical fluxes

LISFLOOD hydrological model

The European Commission Joint Research Centre Institute for Environment and Sustainability, or more briefly, the JRC, devised the LISFLOOD hydrological model [De Roo et al. (2000) and Van der Knijff et al. (2010)]. LISFLOOD is a GIS-based hydrological rainfall-runoff-routing model that is capable of simulating the hydrological processes that occur in a catchment. The specific development objective was to produce a tool that can be used in large and transnational catchments for a variety of applications, including flood forecasting, and assessing the effects of river regulation measures, land-use change and climate change. Outputs from this model were intended to be included in our work on model intercomparisons. Regrettably, they had trouble setting up the model in Africa, because their ground validation was not complete before we finished this part of the study.

NEO DRYMON-SWI profile Soil Moisture

NEO B.V. is a geo-service provider, specializing in monitoring using satellite and aerial imagery, lidar, radar, etc. NEO checks, fixes and updates location based information. In the Netherlands they do this for large scale topography in the transition from line maps to object maps and information models as CityGML. NEO provides the DRYMON services on Soil Moisture derived from radar satellite imagery, provided globally on a daily basis. Derived products are associated with crop yield, droughts, floods, forest fires, water balance models, etc. NEO runs a number of web services in the provision of recent high resolution satellite imagery. The remote-sensing derived NEO profile Soil Moisture estimate DRYMON-SWI (De Lange et al., 2008) is a temporally filtered version of the EUMETSAT ASCAT SWI product. With the ASCAT sensor(s) on board the operational METOP satellite(s) a data supply of surface Soil Moisture is guaranteed by EUMETSAT at (a minimum) until 2020.

This stable situation makes further investments possible and NEO developed its DRYMON product: a global and daily profile Soil Moisture over a depth of 1m in near real time. Within the framework of GLOWASIS the DRYMON product is adapted so that it now also describes the profile Soil Moisture for specific soil compartments that are dimensionally defined by the sizes (depth and spatial resolution) of the upper stores of the hydrological models used in GLOWASIS. They characterize the error in the profile Soil Moisture by applying an error propagation scheme of the uncertainties in the surface Soil Moisture, and by the additional error source caused by the profile Soil Moisture model used. The derived profile Soil Moisture is validated with in-situ measurements of the International Soil Moisture Network (ISMN).

TUWIEN Surface Soil Moisture

The Institute of Photogrammetry and Remote Sensing of the Technical University of Vienna (IPF TUWIEN) conducts research and education in the fields of photogrammetry and remote sensing. It has research groups dealing with geometric modelling (Norbert Pfeifer), physical modelling (Wolfgang Wagner), and image processing (Josef Jansa). The groups cooperate closely to fully utilize technological capabilities offered by photogrammetry and remote sensing to provide essential geo-spatial data for environmental and societal applications (flood warning, urban developments, climate change impacts etc.). They use the scatterometers onboard the ERS and the METOP satellites for global monitoring of Soil Moisture, the Advanced SAR (ASAR) instrument on board the ENVISAT satellite for mapping wetlands, and the SeaWinds scatterometer on board QuikSCAT for freeze-thaw monitoring in high-latitude regions.

The TUWIEN SSM estimate is compiled by combining active microwave (ASCAT – change detection relative Surface Soil Moisture SSM) and passive radiometer (AMSR-E – inverse radiative transfer modelling – VUA/NASA algorithm) information to detect changes in the soil surface layer (about 50 mm) and then infer what the surface Soil Moisture is. The GLOWASIS SSM estimate combines the active and passive estimates at each location (on a 0.25 grid) and time-step using the estimate with the lowest estimated error (for details see Liu et al., 2011).

Rainfall forcing used in the PCR-GLOBWB simulations

PERSIANN [Precipitation Estimation from Remote Sensing Information using Artificial Neural Network] is the product of the Centre for Hydrometeorology and Remote Sensing at University of California, Irvine, led by Soroosh Sorooshian. This bias corrected PERSIANN precipitation estimate maintains total monthly precipitation estimates, consistent with the GPCP (Global Precipitation Climatology Project) product. The newly released data set retains the spatial and temporal features of precipitation estimates provided by the original PERSIANN algorithm at 0.25-degree spatial and 3-hourly temporal resolution, similarly to TRMM.

TRMM 3B42 rainfall (Huffman et al., 2007) is a gauge corrected version of the TRMM 3B42RT estimates used by HYLARSMET at 3 hourly intervals. The product is a blend of estimates of rainfall from (i) a vertically (downward!) pointing C-band radar on board a satellite following an inclined mid-latitude limited orbit and (ii) information from a collection of Geostationary satellites. The product is available over 0.25° (~25 km) square blocks. Because the Geostationary satellites' rainfall algorithms involve primarily cloud-top temperatures to estimate rainfall, the coastal rainfall over South Africa is underestimated because it is relatively 'warm rain'. The C-band radar of the orbiting satellite should correct this but there are times when it skips certain regions in its programmed swath. TRMM therefore tends to produce drier rainfall estimates than gauges over part of the coastal belt, contrary to what occurs in the higher smoother interior, where the comparison behaves similarly to raingauges, as shown in chapter 4. PERSIANN data probably suffer the same corruption.

The third precipitation forcing used in the PCR-GLOBWB modelling is the **ERA** interim precipitation product of the ECMWF. ERA-Interim is a global data set covering the period 1979-2010.

7.2 The model intercomparisons

For intercomparisons, especially for display in figures, it was felt that it was important to condense the estimates from their various spatial geometries to a common grid. The one chosen was the 0.5° [a scale of approximately 50 km at our latitudes] and is the scale at which the PCR-GLOBWB model is computed. Figure 7.2 shows the differences of the coordinate systems of the major data sources.

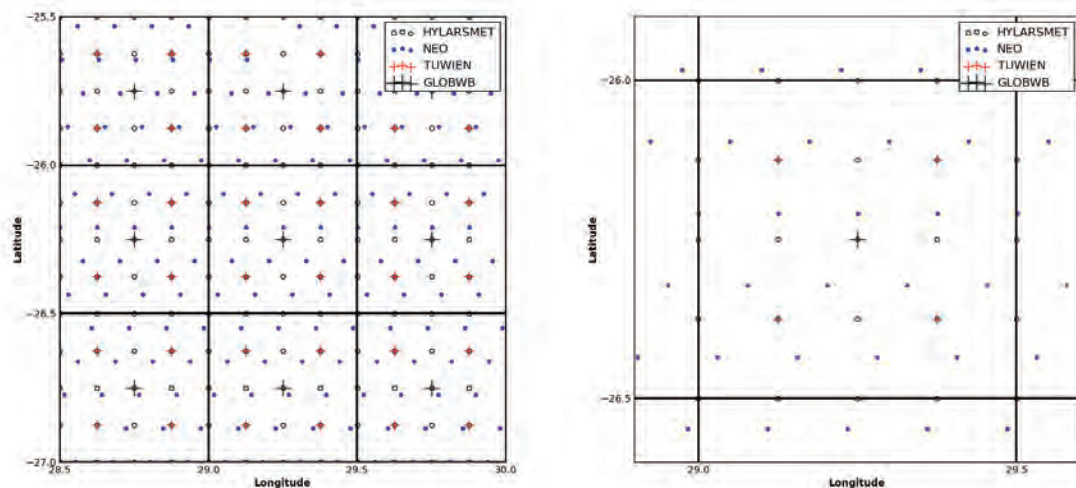


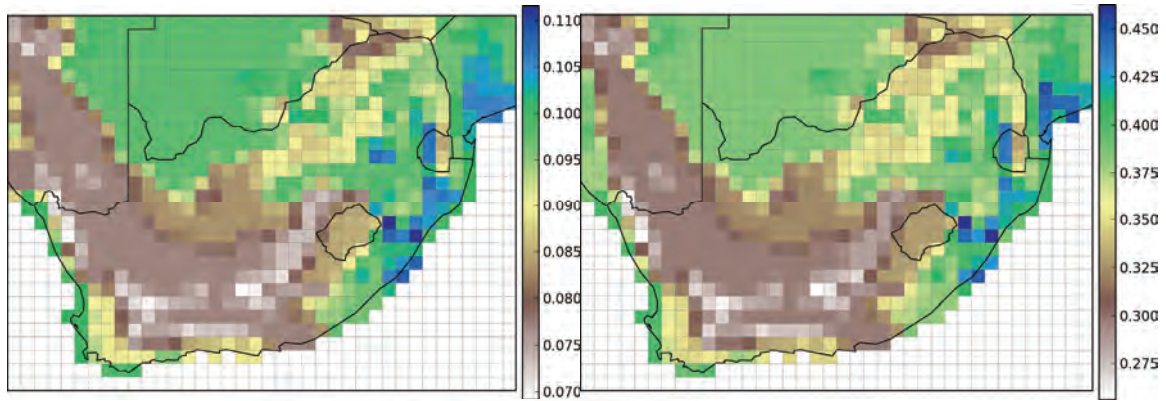
Figure 7.2: Left panel: Grid centre's for each estimate plotted in Lat/Lon. Right panel: Detailed view of a single 0.5° grid square.

To make things a bit more complicated, estimates of Soil Moisture calculated by the PCR-GLOBWB model are available in depth of water in millimetres for each soil layer, so they were carefully converted to SSI equivalents by us. This required contacting the modelling team at the University of Utrecht and obtaining their base sets of soil parameters in order to calculate the available pore space for water storage in the PCR-GLOBWB model's soil layers. Dividing the modelled water depth by the available storage depth at each location and time-step enabled us to produce an SSI from the PCR-GLOBWB simulations.

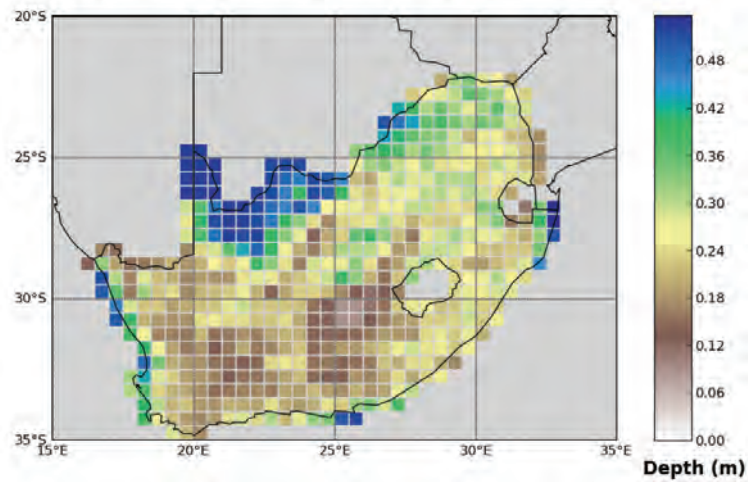
For each cell in the grid and each daily time-step in the 2-year comparative analysis period, spatially averaged estimates were obtained within the 0.5° grid, for each pair of estimates (in this study it was always HYLARSMET vs. other). From these ingredients, we were able to construct spatially coherent and temporally matching SM estimates of all candidate products in the same variable: SSI (our Soil Saturation Index). This necessitated interpreting the SM model outputs based on the other models' Soils Maps. This work enabled us to find temporally co-located daily estimates and generate inter-model scatter-plots to compute R^2 for each 0.5° pixel. The intercomparisons which follow are principally pictorial and cover the region of RSA.

Figure 7.3 shows maximum Soil Water storage averaged over the areas modelled by two models: (i) PCR-GLOBWB which has two soil stores, the depth of upper being approximately 25% of the lower as seen from the legend and (ii) HYLARSMET, which has a single soil store. The sources of these 2 sets of data are different (i) the FAO set and (ii) the RSA set of Middleton et al. (2009) set.

Their spatial scales are also different, (i) 50 km and (ii) 12 km, however, broadly speaking they are reasonably similar, especially when one compares the RSA set spatially averaged over 50 km.



HYLARSMET soil storage capacity



HYLARSMET soil storage capacity

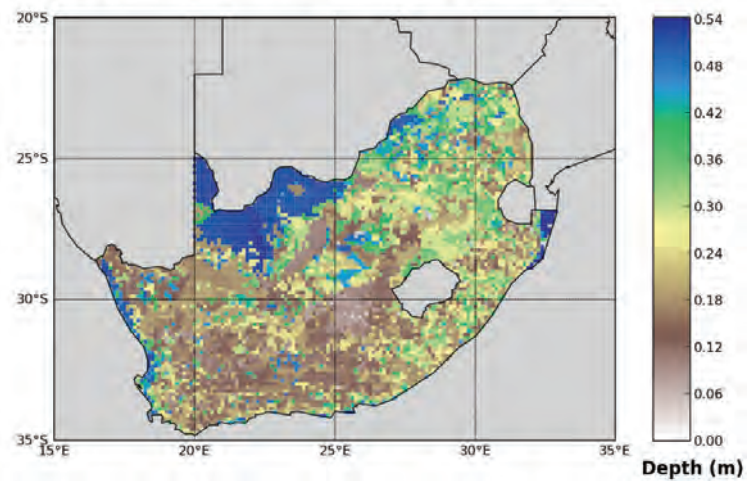


Figure 7.3: Top left panel: Maximum water storage capacity (m) for each grid cell in the upper soil layer of the PCR-GLOBWB model (used to convert the depth of soil water provided into SSI); Top right panel: The same, but for the lower soil layer of the PCR-GLOBWB model; Middle panel: The same for the single store HYLARSMET model; all three at 0.5° spacing; Bottom panel: The same for the HYLARSMET model at the original 0.125° spacing.

Figure 7.4 contains 6 images of SSI estimates averaged over the relevant space-scales and over the same day: 17 Feb 2009. For the PCR-GLOBWB model, we compute the SSI for the upper layer soil store.

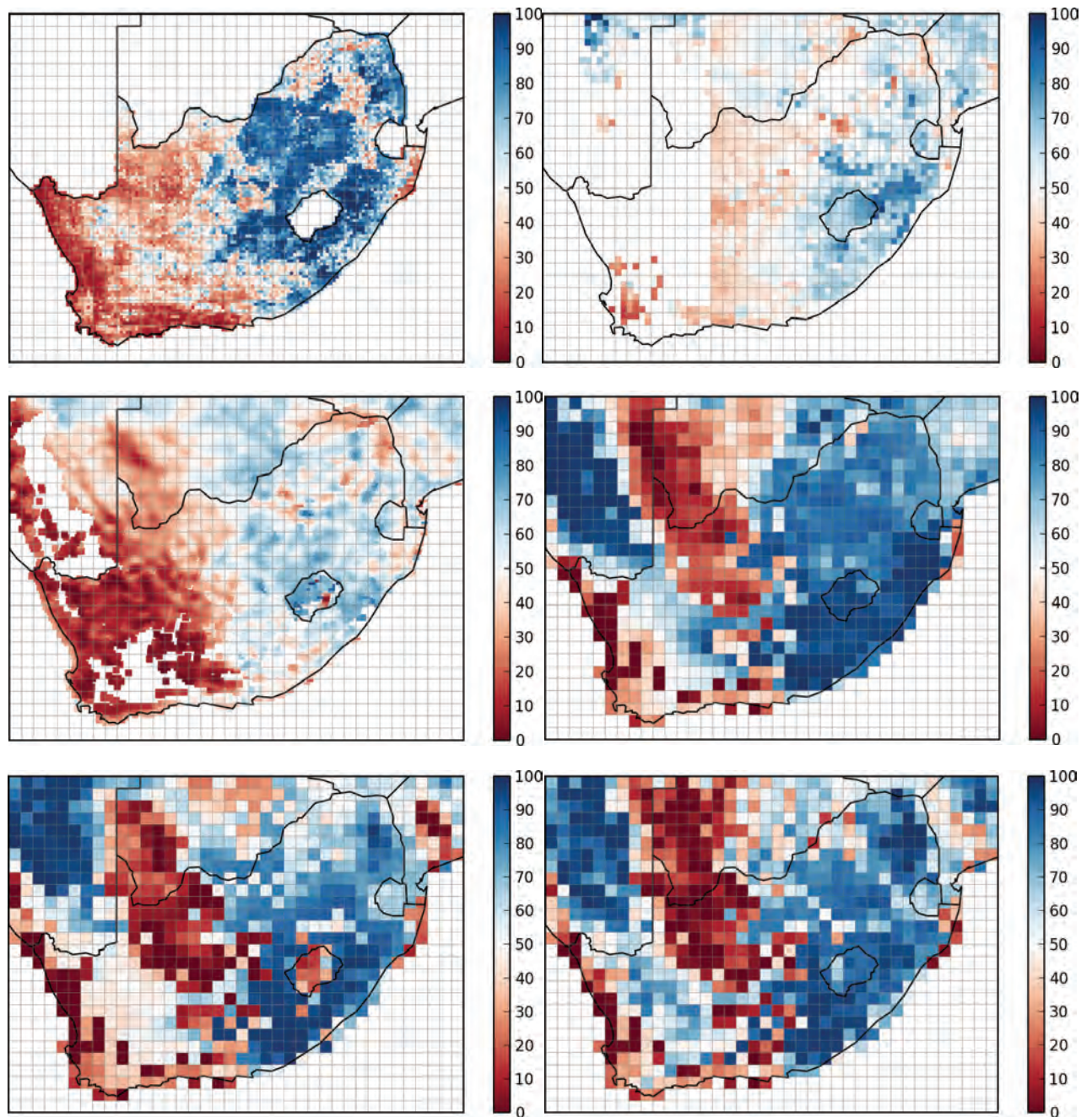


Figure 7.4 From top left by row to bottom right, all SSI for 2009/2/17 with the 0.5° grid overlaid.

- (a) HYLARSMET, forced by TRMM 3B42RT.
- (b) TUWIEN SSM (combined ASCAT/AMSR-E Soil Moisture).
- (c) NEO profile Soil Moisture DRYMON-SWI.
- (d) PCR-GLOBWB SSI with ERA interim rainfall forcing – soil layer 1.
- (e) PCR-GLOBWB SSI with PERSIANN rainfall forcing – soil layer 1.
- (f) PCR-GLOBWB SSI with TRMM 3B42 rainfall forcing – soil layer 1

Figure 7.5 shows SSI values on a selected day, from three runs of PCR-GLOBWB SSI for the second (lower and deeper) soil layer, together with a map of the TUWIEN estimate of the SSM noise.

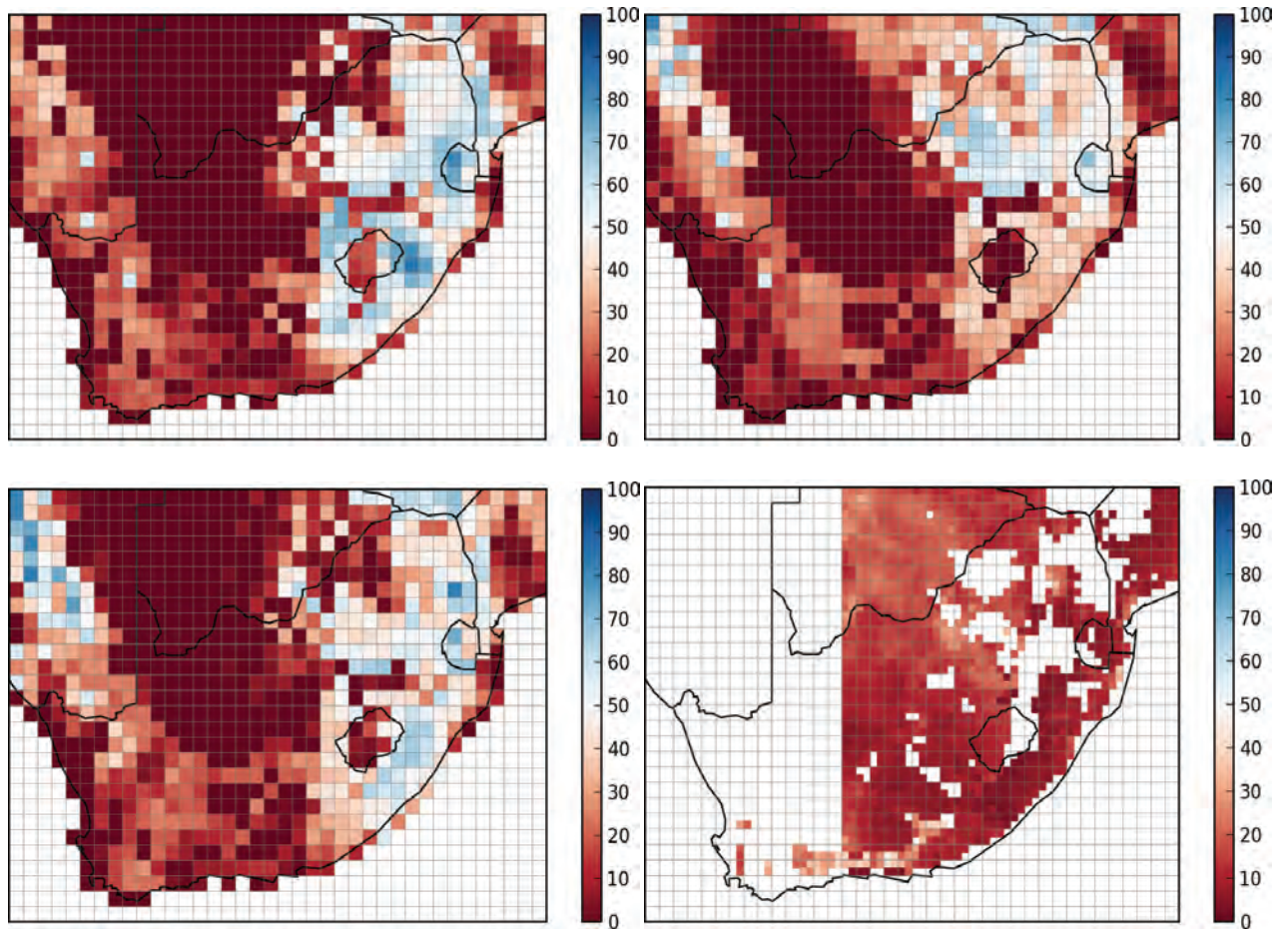


Figure 7.5 From top left by row to bottom right, all SSI for 2009/2/17 with the 0.5° grid overlaid.

- (a) PCR-GLOBWB SSI (ERA forcing – soil layer 2)
- (b) PCR-GLOBWB SSI (PERSIANN forcing – soil layer 2)
- (c) PCR-GLOBWB SSI (TRMM forcing – soil layer 2)
- (d) TUWIEN SWI noise estimate (combined ASCAT/AMSR-E)

The TUWIEN SWI noise estimate on this day (except where there are misses) ranges from 10% to 30%. The error (standard deviation) is of the same order or higher than the SSI values, which accounts for TUWIEN's poor performance when compared to HYLARSMET in the last map of the regressions in Figure 7.8 (h). Strangely, the gaps in Figure 7.5 (d) tend to coincide with those areas where PCR-GLOBWB SSI is relatively high in Figures 7.5 (a) to (c).

In the next set of images in Figure 7.6, we compare daily SSI values computed with HYLARSMET and the other models at a selected cell over the full 2-year period. In the first three rows we have arranged PCR-GLOBWB's upper soil layer 1 on the left and the lower (deeper) soil layer 2 on the right, to aid visual comparison. The images in the last row show how the NEO and TUWIEN estimates fare. The HYLARSMET trace is the same in all images in Figure 7.6.

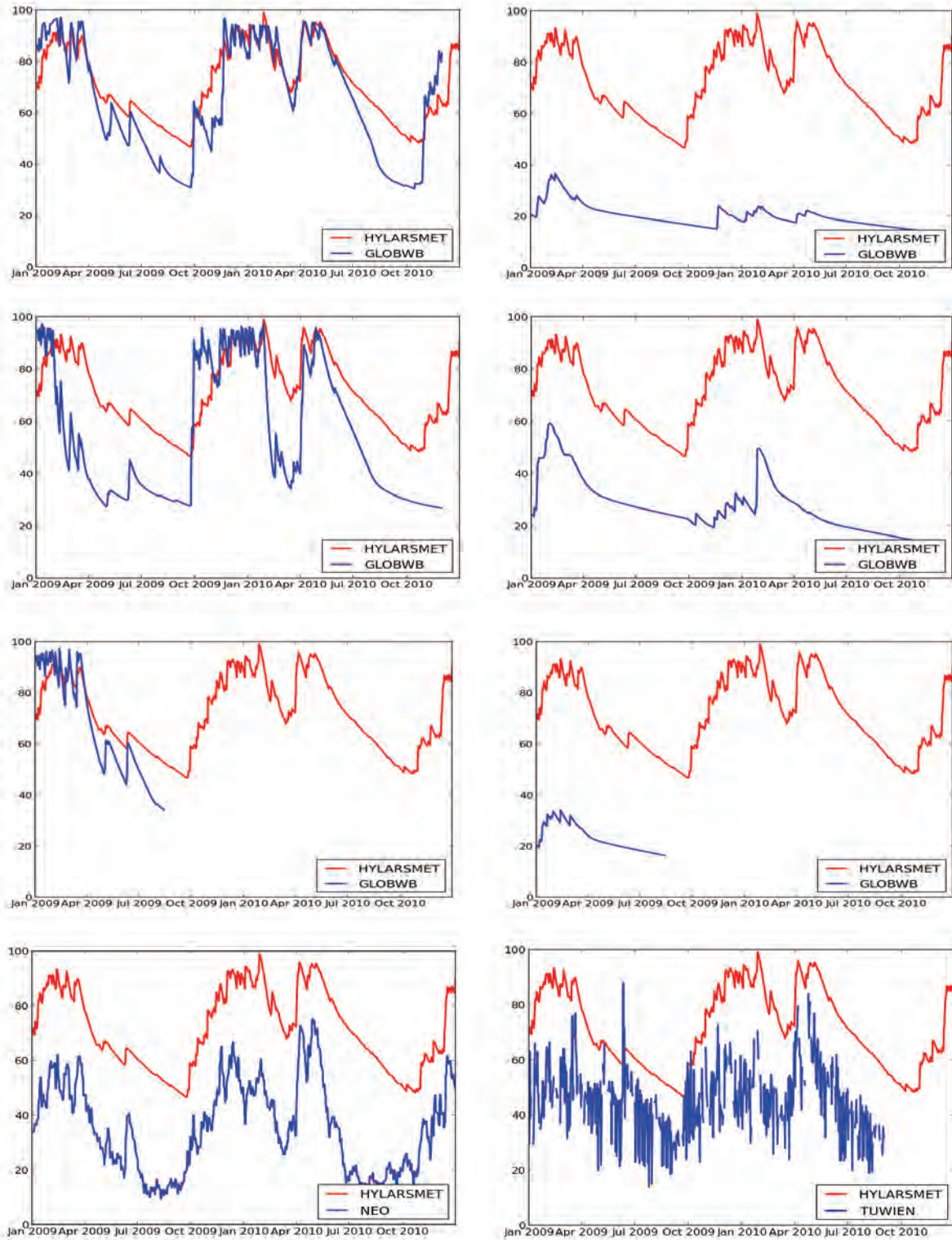


Figure 7.6. Sample daily time-series comparing HYLARSMET and other models over 2 years. In the top three rows PCR-GLOBWB's upper (layer 1) on left and lower (layer 2) store on right. Row 1 ERA forced, row 2 PERSIANN forced, row 3 TRMM forced. In the last row on left NEO profile SM and on right TUWIEN SSM (obviously unfiltered)

In Figure 7.7 the bivariate scatter plots between the HYLARSMET daily estimates of SSI are plotted against the PCR-GLOBWB model's six sets of comparisons in the upper six images of Figure 7.6.

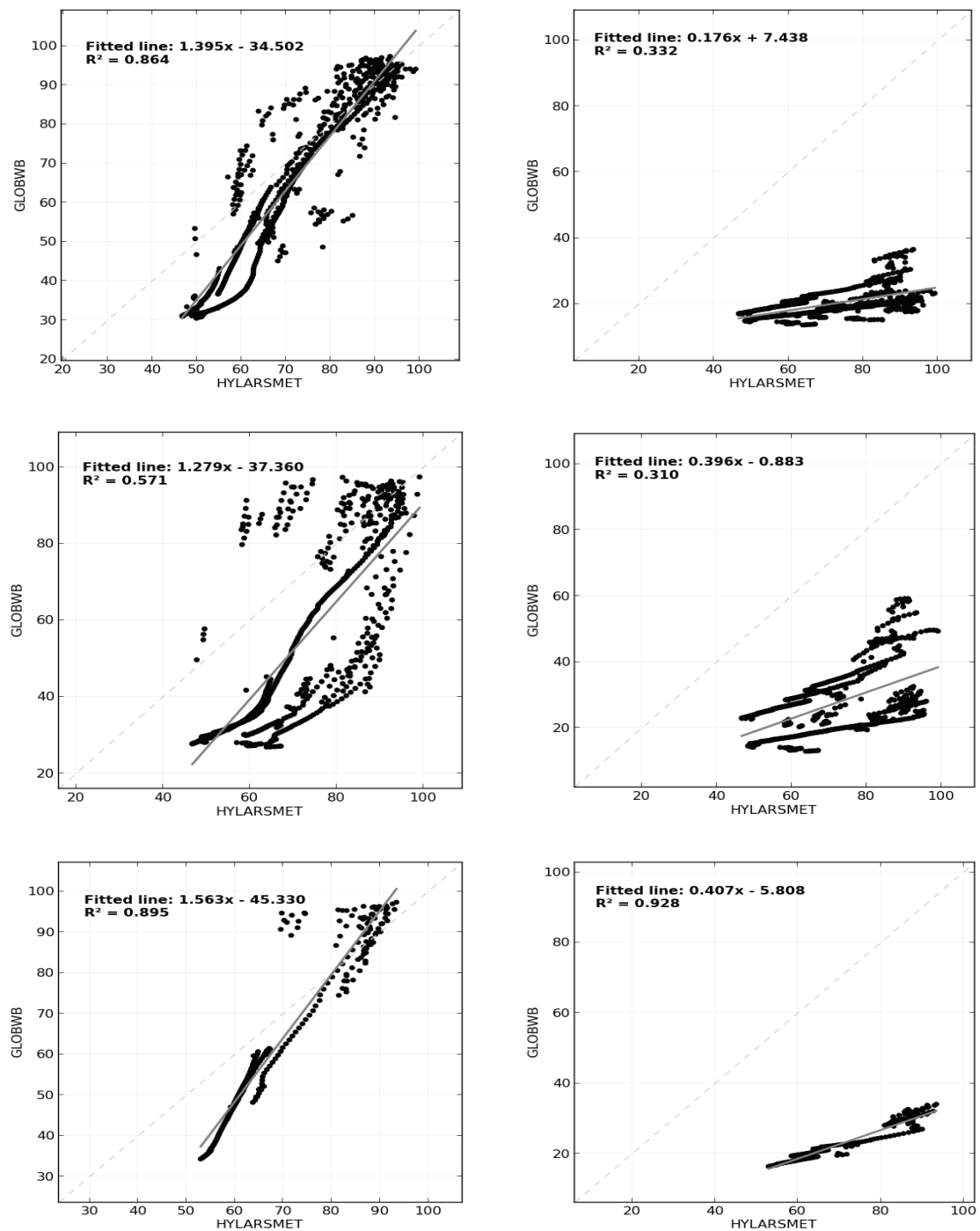


Figure 7.7: Scatter-plot comparing HYLARSMET and PCR-GLOBWB daily values in six images reflecting the behaviour in the upper six images of Figure 7.6. In the three rows PCR-GLOBWB's upper (layer 1) on left and lower (layer 2) store on right. Row 1 ERA forced, row 2 PERSIANN forced, row 3 TRMM forced.

The high correlations of the bottom row might be explained by the fact that the comparisons are limited to the drying out period in the first year of TRMM forcing, whereas the other plots reflect the behaviour over multiple seasons

The R^2 values for all the available 0.5° pixels of the eight intercomparisons following Figure 7.6 appear in Figure 7.8 following.

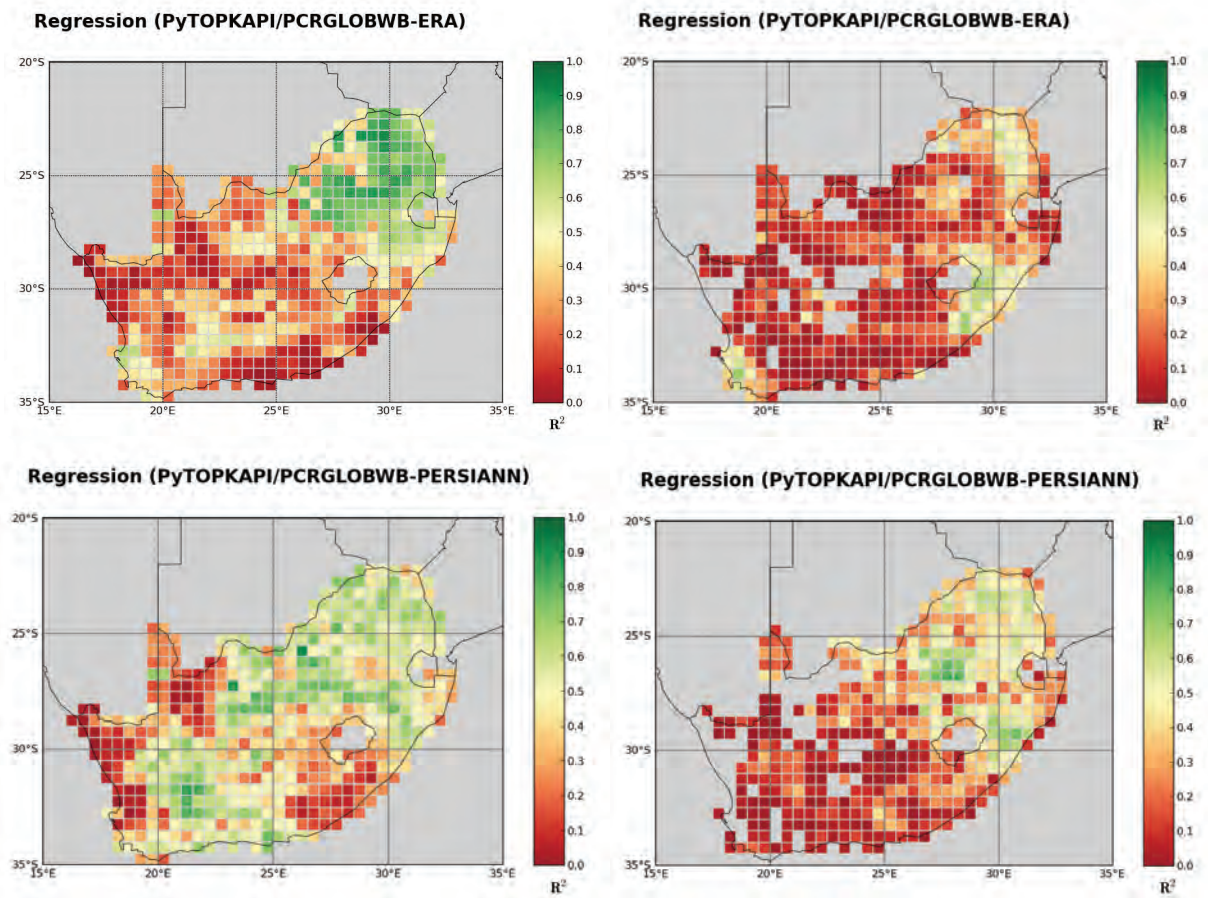


Figure 7.8a: Map of R^2 computed for each 0.5° grid cell (HYLARSMET versus other). Grey areas due to no data from one (or both) of the data-sets during the study period, or due to at least one of the data-sets having a value of zero throughout the period. Soil layer 1 on left and 2 on right.

Top row: (a) and (b) HYLARSMET versus PCR-GLOBWB ERA.

Bottom row: (c) and (d) HYLARSMET versus PCR-GLOBWB PERSIANN rainfall

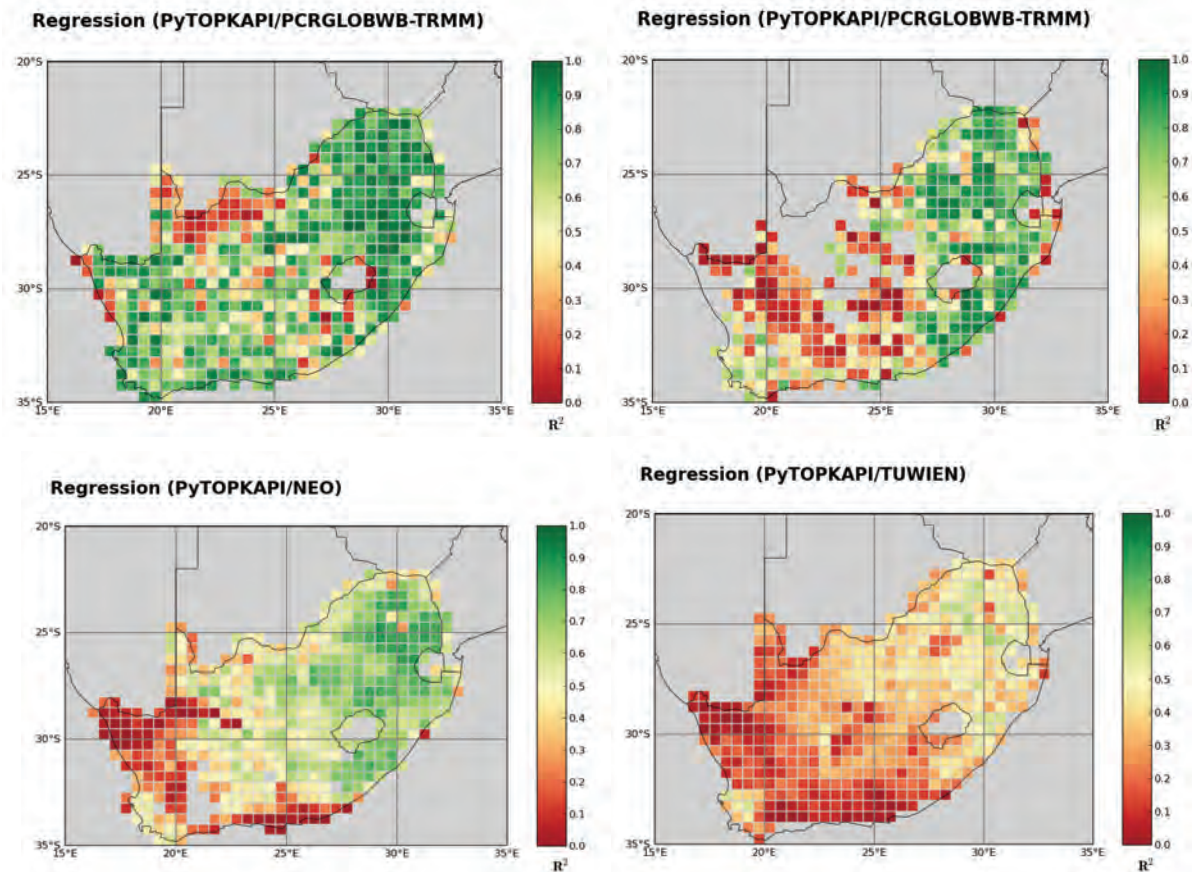


Figure 7.8b: Map of R^2 computed for each 0.5° grid cell (HYLARSMET versus other). Top row: (e) and (f) HYLARSMET versus PCR-GLOBWB TRMM; Soil layer 1 on left and 2 on right. Bottom row: (g) and (h) HYLARSMET versus NEO profile on left SM and TUWIEN SSM on right

7.3 Summary

In this chapter, HYLARSMET estimates of Soil Moisture (SM) computed 3-hourly at 7300 sites in RSA, then accumulated to daily averages, have been compared to a suite of SM products emanating from an FP7 European Union project called GLOWASIS. These HYLARSMET estimates of SM are indicated by the Soil Saturation Index (SSI) introduced in the precursor WRC project K5/1683. To match the different products meaningfully, all SM estimates of the GLOWASIS models were first converted into variables which mimic SSI. To match the timings, all estimates were averaged over each day of 2 years, a period when the estimates were contemporaneous. In addition, because each model has its own spatial discretization, all model estimates were spatially averaged to a common 0.5° block.

This considerable computing effort and organisation of data-streams from disparate sources afforded a meaningful comparison to be made between the outputs of the different models. Each daily 2-year HYLARSMET record averaged on each of the 0.5° blocks was compared against each of the eight model outputs emanating from the GLOWASIS stable, also spatially averaged on the same 0.5° blocks. Some were displayed for visual comparison in the body of the report. Some pairs had their scatterplots displayed as well. All comparative pairs had their scatterplots calculated and the coefficient of determination R^2 calculated for each pairing on each block.

These R^2 values were plotted on maps of South Africa to provide a qualitative feel for the closest model to PyTOPKAPI/HYLARSMET. It turned out that the PCR-GLOBWB TRMM output of Figure 7.8 (e) gave the best overall response, followed by the NEO response in Figure 7.8 (h). As we showed in

earlier work (Pegram et al., 2010; Sinclair and Pegram, 2010), the unfiltered TUWIEN ASCAT/AMSR-E product was noisy (see Figure 7.6, bottom right panel) so that those correlations were disappointingly low.

There are two main inferences that can be drawn from these results. The first is that the models forced by TRMM (both HYLARSMET and PCR-GLOBWB) behave very much the same giving high values of R^2 . On reflection this is not surprising, because SM is strongly affected by rainfall, and even if TRMM is not a replica of the gauge rainfall, as long as it is used unchanged in two similar models, we are likely to get similar results. Encouragingly, this similarity reinforces another observation apparent from Figure 3 and Figure 8 (e). At the scale considered, the FAO soil properties are similar to those we have carefully determined in South Africa [Middleton and Bailey, (2009)].

This last observation gives hope that we can use PyTOPKAPI in other places in Africa, where TRMM and FAO data are available. In addition, because we are cautiously confident of the robustness of the modelling procedure, we should be able to provide useful SM estimates on a daily basis, where none presently exist. These are useful because drought monitoring, water resources modelling and risk determination depend crucially on the Soil Moisture state of a region.

8. Maintenance of HYLARSMET community & HYLARSMET Outreach

Preamble

The outreach through workshops, lectures, presentations and papers are summarised in this chapter. The following sections showcase our activities outside the workrooms of our offices and computers. We have made a concerted effort to “get HYLARSMET out there”. This has been done by forming personal contacts in the flesh and on the internet, ranging from small group discussions to presentations and workshops through to presentations at National and International Symposia and Congresses, then by dissemination through papers in International Journals. We start by reporting on the work involved in establishing the open source model available on the chosen website by the new name PyTOPKAPI.

8.1 PyTOPKAPI

This section provides evidence that the objectives have been met and provides our rationale for the choices made in order to meet these objectives. The primary goal was to put in place some structure that decreases the likelihood of the PyTOPKAPI model falling into disuse and the code becoming outdated. The investment in the model development by the WRC (Pegram et al., 2010) would be wasted if this were to occur. Our belief is that by making the model as easy as possible to obtain and use, there will be greater uptake, and by releasing the model code under an open source license we enable contributors to improve the model and adapt it to their own needs.

PyTOPKAPI implements the TOPKAPI model (Lui and Todini, 2002) as an extension package for the Python programming language (<http://www.python.org>). Python is well suited for use in science and engineering as it allows for the convenient management of large data arrays and because it can be extended and interfaced with many other existing languages used by practitioners in the field (FORTRAN, C, C++, etc.), in order to take advantage of existing well tested numerical code. The PyTOPKAPI 0.2.0 release has been listed on the Python Package Index (PyPI), which is an online searchable directory of add-on Python packages.

The section comprises the following headings (web addresses for access are also given for convenience):

PyTOPKAPI documentation

<http://sahg.github.com/PyTOPKAPI/>

Public source code repository and issue tracking

<http://github.com/sahg/PyTOPKAPI/>

Mailing list

<http://sahg.github.com/PyTOPKAPI/contact.html/>

Figure 8.1 shows an overview of the PyTOPKAPI web presence. The project home page, source code repository, bug tracker and download area are hosted by Github. The project mailing list and mailing list archives are hosted by Google groups. The PyTOPKAPI entry on PyPI is a good way to increase the visibility of PyTOPKAPI as people searching for useful Python packages will most likely look there first. The project listing on PyPI points to the PyTOPKAPI home page and download area hosted at Github.

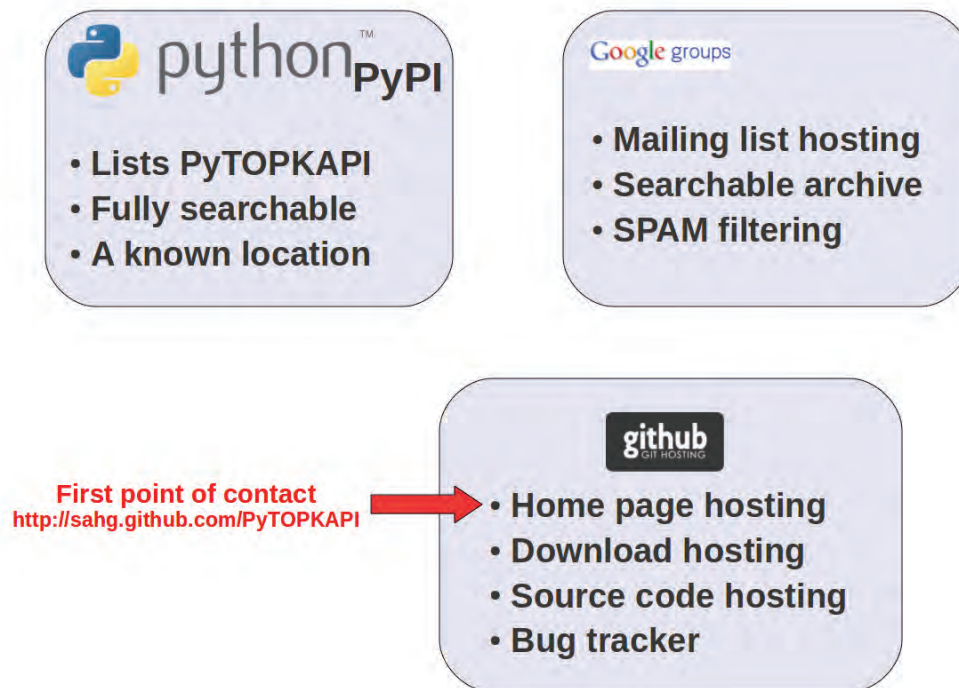


Figure 8.1: Overview of the PyTOPKAPI web presence. PyPI increases the visibility of PyTOPKAPI as most Python users will look there when searching for software. Github hosts the home page, source code repository and download area. Google groups hosts the mailing list and mailing list archive.

Figure 8.2 shows the PyTOPKAPI entry on PyPI, this page provides information on where to find documentation for PyTOPKAPI as well as a link to the package download page.

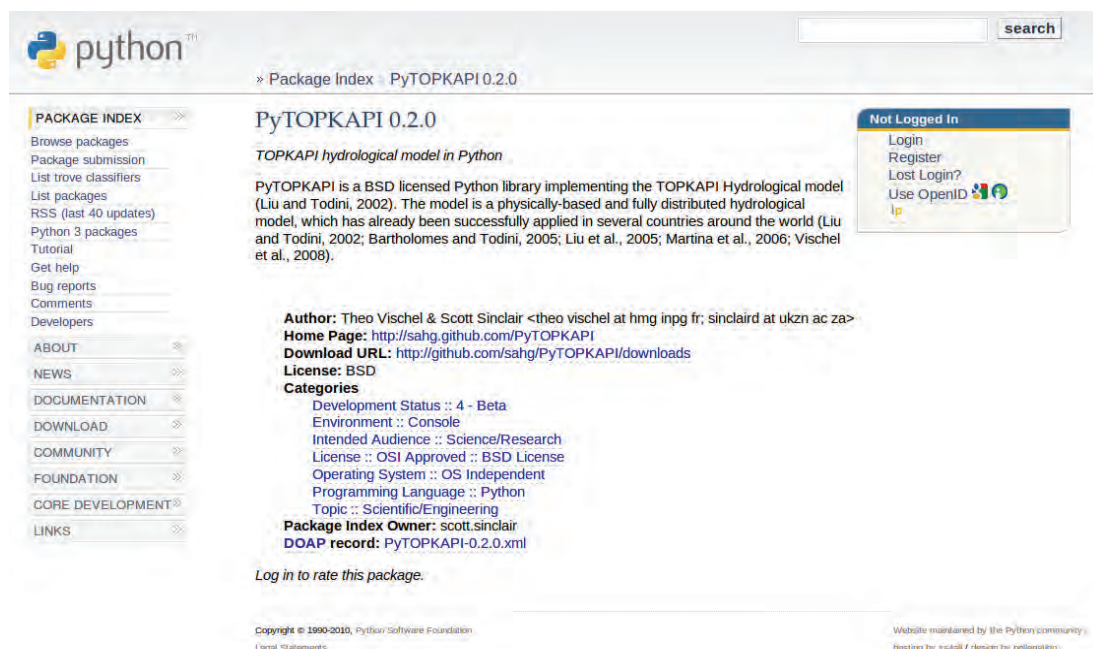


Figure 8.2: Screenshot of the PyTOPKAPI listing on the Python Package Index (<http://pypi.python.org/pypi/PyTOPKAPI/>). The PyPI is a fully searchable online directory of software packages (add-ons) available for the Python language and contains approximately 11500 packages (as at 1 October 2010).

8.1.1 PyTOPKAPI documentation

The PyTOPKAPI documentation is primarily available online, but will also be made available as a PDF document in future to enable offline viewing. Figure 3 shows a screenshot of the PyTOPKAPI documentation front page. The website can be viewed online at the web address given in the caption of the figure. This documentation contains basic information on how to use the PyTOPKAPI package and describes its features.



Figure 8.3: Screenshot of the PyTOPKAPI documentation website: <http://sahg.github.com/PyTOPKAPI/>. The website will be periodically updated throughout the life of the project.

The PyTOPKAPI package is released in two forms. The first is a Python source code distribution, which can be installed on any operating system where Python works, while the second is a binary installer for 32-bit Windows. A 64-bit Windows installer has not been produced at this stage but will be once there is a demand. Figure 8.4 shows a screen capture of the PyTOPKAPI download page, indicating that there were 57 downloads of the main program (see red box) within 2 days after the release of version 0.2.0. This is exciting and is particularly gratifying for the project. An example catchment setup is also

available to allow potential users of the model to follow the introductory tutorial and 55 copies of that were downloaded in the same period.

name	Description	Uploaded	Downloads	Size
PyTOPKAPI-Example-0.2.0.zip	PyTOPKAPI Example catchment setup	2010-09-21	55	1.2MB
PyTOPKAPI-0.2.0.win32.exe	PyTOPKAPI 0.2.0 (32-bit Windows installer)	2010-09-21	2	235KB
PyTOPKAPI-0.2.0.zip	PyTOPKAPI 0.2.0 (zip)	2010-09-21	57	45KB
PyTOPKAPI-0.2.0.tar.gz	PyTOPKAPI 0.2.0 (tar.gz)	2010-09-21	59	36KB

Tag	Download	Description	Date	Commit
version0.1.1	tgz zip	Tag release 0.1.1 of the model.	2008-09-08	4de3d7d
0.2.0	tgz zip	Version 0.2.0 release	2010-09-20	2595ffb

Figure 8.4: Screenshot of the PyTOPKAPI software download page (<http://github.com/sahg/PyTOPKAPI/downloads/>). The current model release is version 0.2.0. The red box highlights the number of downloads within 2 days of the PyTOPKAPI release.

8.1.2 Public source code repository and issue tracking

The PyTOPKAPI source code has been stored in a version control system since very early on in the development effort. Recently the repository format has been converted to a very powerful distributed version control system called Git (<http://git-scm.com/>). There are a number of options that provide hosting of Git repositories accessible via the internet, we selected Github as a hosting service mainly due to its popularity and user friendly tools. Figure 5 shows a typical screenshot of the online repository view that would greet a visitor to the PyTOPKAPI Github page. It is not necessary for regular users to understand or interact with the Git repository; this is a tool for collaborative model development. Detailed instructions for using Github in the context of the PyTOPKAPI project are available in the PyTOPKAPI documentation.

The screenshot displays the GitHub interface for the `sahg / PyTOPKAPI` repository. At the top, the GitHub logo and navigation links (Home, Pricing and Signup, Training, Gist, Blog, Login) are visible. The repository name is prominently displayed, along with options to Watch, Fork, and Download Source. Below this, a tabbed interface shows the 'Source' tab selected, with links to Commits, Network (2), Issues (2), Downloads (6), and Graphs. The current branch is 'develop'.


The commit history table shows the following entries:

name	age	message	history
<code>.gitignore</code>	July 29, 2010	ENH: Add .gitignore [scottza]	
<code>LICENSE</code>	September 02, 2010	DOC: Update license file [scottza]	
<code>README</code>	August 31, 2010	DOC: Update project README [scottza]	
<code>TOPKAPI_Example/</code>	August 17, 2010	REF: Convert remaining DOS line endings to Unix... [scottza]	
<code>build_and_test.sh</code>	April 02, 2009	Force a clean rebuild for testing. [scottza]	
<code>change_log</code>	July 21, 2010	REF: Convert DOS to Unix line endings [scottza]	
<code>create_distributions.py</code>	November 06, 2009	Update create distributions script [scottza]	
<code>docs/</code>	1 day ago	DOC: Add note on gitwash to the developer docs [scottza]	
<code>pytopkapi/</code>	September 22, 2010	DOC: Update API documentation to standardize di... [scottza]	
<code>setup.py</code>	1 day ago	DOC: Update information for PyPI [scottza]	
<code>tests/</code>	August 17, 2010	REF: Convert remaining DOS line endings to Unix... [scottza]	


The README section below the table describes PyTOPKAPI as a Python implementation of the TOPKAPI Hydrological model, licensed under BSD. It mentions that the model is physically-based and fully distributed, and has been successfully applied in several countries.

Figure 8.5: Screenshot of the PyTOPKAPI source code repository hosted on Github (<http://github.com/sahg/PyTOPKAPI/>). The latest development version of the model can easily be downloaded, or the source code can be browsed online. For users with more sophisticated requirements, a local version of the code repository can be downloaded and easily updated to reflect the latest official developments.

The following screen capture shows the history of material alterations to the code, so people making enquiries can identify the most recent update and what was done.



[Home](#)
[Pricing and Signup](#)
[Training](#)
[Gist](#)
[Blog](#)
[Login](#)


[sahg / PyTOPKAPI](#)

[Watch](#)
[Fork](#)
[Download Source](#)

1
 2

[Source](#)
[Commits](#)
[Network \(2\)](#)
[Issues \(2\)](#)
[Downloads \(6\)](#)
[Graphs](#)




Branch: develop

[Switch Branches \(3\)](#)
[Switch Tags \(2\)](#)
[Comments](#)
[Contributors](#)




[PyTOPKAPI / Commit History](#)

Keyboard shortcuts available

2010-09-29

DOC: Add note on gitwash to the developer docs	commit c493f1d68650b61dd7bc tree d5df0082d1f511273ceb parent cf15f35ad2b6a4eb95c2
 scottza (author) 1 day ago	
REL: Start basic notes on the release management process	commit cf15f35ad2b6a4eb95c2 tree bce7155e4f28a7ac86c1 parent 86cbf1bffd9f2bf1539b
 scottza (author) 1 day ago	
DOC: Update information for PyPI	commit 86cbf1bffd9f2bf1539b tree 5dcb8da985bf03ab00fd parent 4c8e7024ec33c07a741a
 scottza (author) 1 day ago	

2010-09-28

DOC: Label docs as docs for release 0.2.0	commit 4c8e7024ec33c07a741a tree 7d662db59208cc690b42 parent f7f7bf0972cf14e15e12
 scottza (author) 2 days ago	
DOC: Adjust the documentation structure to get the correct page order	commit f7f7bf0972cf14e15e12 tree a586e80be0afd47c0c4 parent ba6c018bf1fa58af4b70
 scottza (author) 2 days ago	
DOC: Include developer docs generated by gitwash	commit ba6c018bf1fa58af4b70 tree 9deb003aad76aad94bf8 parent c88aface0c6e6c69a8ce
 scottza (author) 2 days ago	

2010-09-22


DOC: Update API documentation to standardize display of units	commit c88aface0c6e6c69a8ce tree f241d08b6a3f497637d9 parent 2cde5522a3b5af1655e0
 scottza (author) September 22, 2010	

Figure 8.6: Screenshot of the PyTOPKAPI commit history page on Github. This allows easy browsing of all changes to the source code and each code commit to the source code repository can be viewed in detail (see example in Figure 8.7 following).

The screenshot displays the GitHub interface for the `sahgh/PyTOPKAPI` repository. At the top, the GitHub logo and navigation links (Home, Pricing and Signup, Training, Gist, Blog, Login) are visible. Below the repository name, there are tabs for Source, Commits, Network (2), Issues (2), Downloads (6), and Graphs. The 'Commits' tab is selected, showing a list of commits. The current commit is titled 'DOC: Add note on gitwash to the developer docs' by user 'scottza' (author), made 1 day ago. The commit hash is 'c493f1d68658b61dd7bc'. The file 'docs/source/developer.rst' is highlighted, and its content is shown in a diff view. The diff shows changes to the developer documentation, adding a note about gitwash. The page also includes a footer with GitHub's terms of service and Rackspace Hosting logo.

Figure 8.7: An example of an individual code commit for PyTOPKAPI. The changes are shown in Unified Diff format and software patches are typically provided in the same manner.

Bug reports and feature requests can be recorded using the PyTOPKAPI issues page. The issues page (as shown in figure 8) is basically a task list, where each task allows for discussion and the attachment of source code patches. Bug reports and feature requests will be most rapidly dealt with when accompanied by a software patch.

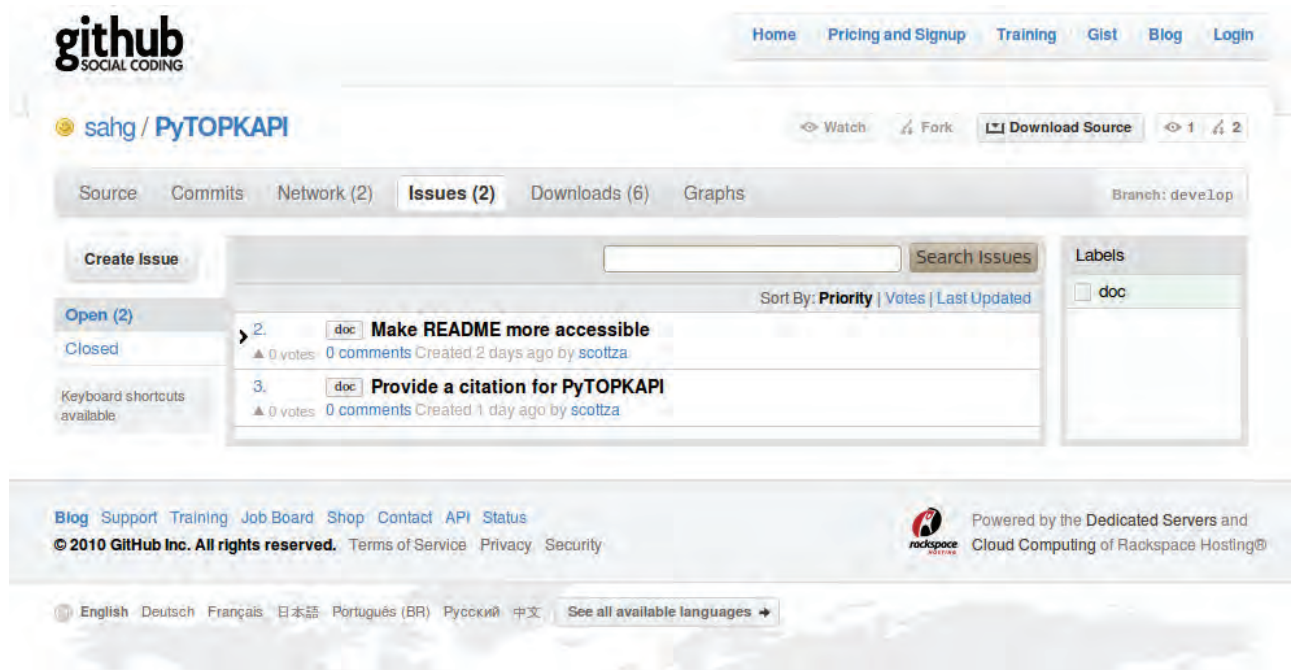


Figure 8.8: A screenshot of the PyTOPKAPI issues page (<http://github.com/sahgh/PyTOPKAPI/issues/>). The issues page is essentially a task list, which helps the developers to prioritize and keep track of Bugs and Feature Requests for PyTOPKAPI.

8.1.3 Mailing list

Public mailing lists are an important tool for recording discussions and questions related to open source projects. E-mails sent to the mailing list address are automatically sent to all of the list subscribers, allowing open communication and sharing the burden of responding to commonly asked questions related to the software. We have chosen to host a mailing list using Google Groups free hosting service. The list is open for anyone to join and further information on how to do so can be found at <http://sahgh.github.com/PyTOPKAPI/contact.html/>. See Figure 8.9 for the opening page.

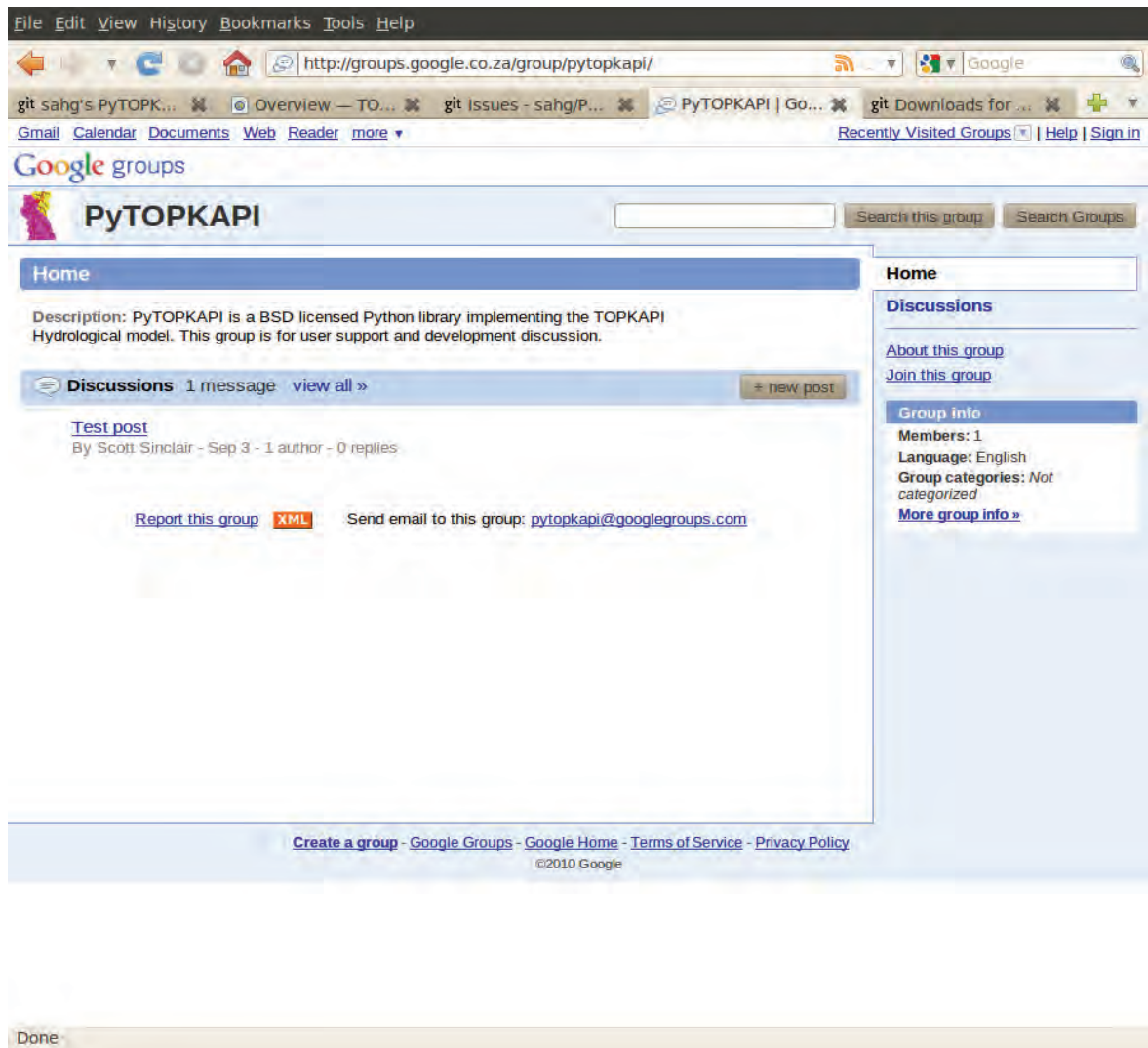


Figure 8.9: A screenshot of the PyTOPKAPI mailing list front page (<http://groups.google.co.za/group/pytopkapi/>). The list is hosted by Google groups and is open for anyone to join. The group has only been recently set up and doesn't have any members yet.

8.1.4 Detail of work carried out to achieve the objectives

The preceding portion of this report provides an overall picture of the final results of the work reported here and the polished exterior hides the amount of work required of the project team in order to achieve the final product. This section of the report attempts to provide the reader with a clearer picture of the tasks that were carried out and the skills which were acquired in order to bring the model to completion in its current form. We note that we can describe a portion of the work with words, but can provide evidence of the other part of the work with transcribed documentation from the web-site.

1. Source code repository

- A review of software version control systems and software development practices was carried out in order to determine which version control system was most suited to use in this project.
- A review of online repository hosting providers was carried out in order to select a suitable hosting service. Factors such as cost, storage space, backup infrastructure, tools provided and popularity were considered.

2. Preparation of the PyTOPKAPI release

- Required significant work on the code base (mainly documentation at that stage) and consideration of changes that would ensure a stable model interface going forward.
- Testing and preparation to ensure that the PyTOPKAPI source code is able to run without changes on multiple computer architectures (e.g. 32-bit, 64-bit), operating systems (e.g. Linux, Windows) and with many different Python versions (currently 2.5.x and 2.6.x).
- Learn how best to distribute and structure the model code to make distribution and installation easy for the user on the platforms mentioned above. Build and test executable installers for Windows users (see PyTOPKAPI download page on Github).
- Actually do the work required to upload the model code and release download packages onto Github.
- Learn how to list the model on the PyPI website and make sure that this is done.
- Research suitable procedures for the release of the software.
- Research and select a suitable open source licence.
- An important aspect is the ONGOING work of fixing bugs, assisting users in getting the model running and updating the documentation and codebase.
- Figure 8.10 shows a sample of the 393-page source code change log.

3. Documentation

- Understand and update/write documentation for all model code, converting it from ‘research’ code into a product which is seamlessly accessible and will not ‘fall over’ when downloaded and run by others.
- Research and select tools and systems to ease the burden of documentation and allow documentation to be written and updated in single locations under version control. This makes it easier to avoid the documentation getting out of date relative to the code.
- Learn how to use the selected tools to generate online documentation. The system selected is Sphinx (<http://sphinx.pocoo.org>).
- Research and then do the actual work of getting the generated documentation posted on the Github hosting site.

Figure 8.10 (which spreads over onto a second page) shows two entries (*changesets*) from the PyTOPKAPI *changelog*. The changelog is produced by interrogating the source code version control system. In figure 10 (in which the text is colour-coded according to the following explanation), each change is separated by a **brief description of the change, the change author’s name, as well as the date and time** that the changes were logged by the source control system. Wherever a line is started with a ‘-’ sign, **a line has been deleted** in the code/documentation. A ‘+’ sign indicates **a line that has been inserted**. Nearby lines of code that have remained unchanged are included to provide context for anyone examining a particular changeset. The first changeset shown in Figure 10 describes the changes required in order to fix a bug in the code. This is indicated by the consistent use of **BUG:** at the beginning of code changes that describe bug fixes. The second changeset in Figure 10 describes the changes made to the PyTOPKAPI codebase in order to provide accurate meta-data for the package listing on PyPI. This is a documentation change as indicated by the use of **DOC:** at the start of the changeset description.

BUG: Make sure versioning info doesn't get lost when building from a source dist

Scott Sinclair, Tue Aug 24 13:17:12 2010 +0200---

setup.py | 9 +++++++-

1 files changed, 8 insertions(+), 1 deletions(-)

diff --git a/setup.py b/setup.py

index 23cb401..4faa0a4 100644

--- a/setup.py

+++ b/setup.py

@@ -13,13 +13,20 @@ MICRO = 0

ISRELEASED = False

VERSION = '%d.%d.%d' % (MAJOR, MINOR, MICRO)

+dev_version_py = 'pytopkapi/__dev_version.py'

+

def generate_version_py(filename):

try:

if os.path.exists(".git"):

+ # should be a Git clone, use revision info from Git

s = subprocess.Popen(["git", "rev-parse", "HEAD"],

stdout=subprocess.PIPE, stderr=subprocess.STDOUT)

out = s.communicate()[0]

GIT_REVISION = out.strip()

+ elif os.path.exists(dev_version_py):

+ # should be a source distribution, use existing dev

+ # version file

+ from pytopkapi.__dev_version import git_revision as GIT_REVISION

else:

GIT_REVISION = "Unknwn"

except:

@@ -46,7 +53,7 @@ git_revision = '%s'

return FULL_VERSION, GIT_REVISION

if __name__ == '__main__':

- full_version, git_rev = generate_version_py('pytopkapi/__dev_version.py')

+ full_version, git_rev = generate_version_py(dev_version_py)

setup(name='PyTOPKAPI',

version=full_version,

DOC: Update information for PyPIScott Sinclair, Wed Sep 29 13:47:59 2010 +0200--- setup.py | 25

+++++

1 files changed, 24 insertions(+), 1 deletions(-)

diff --git a/setup.py b/setup.py

index ab1f70b..84226d9 100644

--- a/setup.py

+++ b/setup.py

@@ -57,12 +57,35 @@ if __name__ == '__main__':


```

setup(name='PyTOPKAPI',
      version=full_version,
-     description='SAHG TOPKAPI model implementation',
+     description='TOPKAPI hydrological model in Python',
+     long_description = """\
+PyTOPKAPI - a Python implementation of the TOPKAPI Hydrological model
+=====
+
+PyTOPKAPI is a BSD licensed Python library implementing the TOPKAPI
+Hydrological model (Liu and Todini, 2002). The model is a
+physically-based and fully distributed hydrological model, which has
+already been successfully applied in several countries around the
+world (Liu and Todini, 2002; Bartholomes and Todini, 2005; Liu et al.,
+2005; Martina et al., 2006; Vischel et al., 2008).
+
+""",
      license='BSD',
      author='Theo Vischel & Scott Sinclair',
      author_email='theo.vischel@hmg.inpg.fr; sinclaird@ukzn.ac.za',
+     url='http://sahg.github.com/PyTOPKAPI',
+     download_url='http://github.com/sahg/PyTOPKAPI/downloads',
      packages=['pytopkapi',
                'pytopkapi.parameter_utils',
                'pytopkapi.results_analysis'],
+     classifiers=[
+         'Development Status :: 4 - Beta',
+         'License :: OSI Approved :: BSD License',
+         'Environment :: Console',
+         'Operating System :: OS Independent',
+         'Intended Audience :: Science/Research',
+         'Programming Language :: Python',
+         'Topic :: Scientific/Engineering',
+     ],
)

```

Figure 8.10: Example entries from the PyTOPKAPI changelog. The first entry shows the source code changes that corrected a bug and the second entry indicates the additional meta-data added to the existing codebase in order to accurately describe PyTOPKAPI on the Python Package Index (PyPI).

8.1.5 Summary of section

This section has provided evidence, through publicly accessible links, that source code hosting, bug tracking, a mailing list and a website have been set up and are currently functioning. A particularly gratifying surprise was the number of downloads recorded within 2 days of the launch of the suite. As the original report was being finalised, 25% more downloads had been recorded. This was exciting, as it exceeded our hopes for effective outreach. An ongoing task will be to encourage the participation of new users through our collaborations with local and international researchers. The sites were particularly useful when we conducted the workshop.

8.2 Workshops and Presentations

Other than the maintenance of software, the most important activity for us was the PyTOPKAPI workshop held in the WRC headquarters in Pretoria on 13 February 2012. This required an intensive activity in automating the code which had been developed and reported here, making it palatable for potential users. It had to be able to work on a wide variety of computer platforms, which was a taxing task. It was solve by ensuring that the package would run on its own operating system, to be installed on each attendee's machine. There were 21 attendees and the workshop lasted the better part of the day. The people who came to the workshop were:

name	email
Allan Bailey	AllanB@ssi.co.za
Andre Gorgens	Andre.Gorgens@aurecongroup.com
Brian Jackson	jacksonb@inkomaticma.co.za
Eddie Riddell	edriddell@gmail.com
Eugene Poolman	Eugene.Poolman@weathersa.co.za
Evison Kapangaziwiri	evisonk@gmail.com
Gabriel Lekalalala	lekalakalaRG@agric.limpopo.gov.za
Jean-Marc Mwenge Kahinda	JMwengeKahinda@csir.co.za
Jenny Pashkin	pashkinj@dwa.gov.za
Johan Malherbe	Johan@arc.agric.za
Malin Govender	malin@escience.co.za
Mark Summerton	Mark.Summerton@umgeni.co.za
Michael Weston	michael@escience.co.za
Nobuhle Majozi	nobuhle.majozi@gmail.com
Paulo Kagoda	pkagoda@gmail.com
Simon Chambert	Schambert@golder.co.za
Simon Ngoepe Malose	NgoepeM@dwa.gov.za
Stephen Mallory	stephen@waterresources.co.za
Tendai Sawunyama	tendai@waterresources.co.za
Trevor Coleman	TColeman@golder.co.za
Tsholofelo Mbotho	MbothoT@dwa.gov.za

Prof Pegram gave a 90-minute presentation on PyTOPKAPI, its genesis and philosophy, covering some of the technicalities. This was followed by a hands-on workshop run by Dr Sinclair, who started by handing out 21 flash-drives with 4Gb capacity. Each of these contained an Ubuntu operating system and the completed PyTOPKAPI software set. Dr Sinclair spent the rest of the day demonstrating and tutoring the participants in the use of the software. Most found the day demanding but valuable; some enquired when the follow-up workshop would be. The hand-out was a 21 page pdf, appended to this report.

8.3 Other Dissemination and Contact Activities

8.3.1 Continued contributions to ARC's monthly Umlindi report.

This effort requires keeping our data archive functional and up to date. Many of the core data products are received via our EUMETCast reception station, which needs regular attention to ensure that software is kept up to date and the large volume of incoming data is properly deleted, or directed

to the archive as required. We are continually working on the development of tools to streamline/automate the calculations and tests for missing data, as well as tools to aid us in the task of producing the Umlindi reports. For example September 2012:

Average Soil Moisture conditions September 2012

The figures presented below summarize recent results of the countrywide Soil Moisture modelling carried out by the University of KZN Satellite Applications and Hydrology Group. The first three panels show the monthly averaged Soil Moisture conditions for September 2011, September 2012 and August 2012. The colour scale ranging from brown to blue represents the Soil Saturation Index (SSI), which is defined as the percentage saturation of the soil store in the PyTOPKAPI hydrological model. The modelling is intended to represent the mean Soil Moisture state in the root zone, and SSI is controlled by rainfall, evapotranspiration and down-slope drainage of the soil. The figure in the fourth panel shows the month on month SSI difference between September 2012 and August 2012, while the fifth panel shows the year on year SSI difference between September 2012 and September 2011. In both cases the brown part of the colour scale represents drier than previous and the green part wetter than previous modelled Soil Moisture conditions. The month-on-month difference map indicates wetter than previous conditions for September compared to August over KwaZulu-Natal. The remainder of the country is neutral, with a few parts of the Free State, Eastern Cape and Western Cape slightly drier than the previous month. The modelled year-on-year difference map shows generally drier than previous conditions for the Northern Cape, parts of the Western Cape, Northwest, Free State and Limpopo. Wetter than previous conditions are indicated for KwaZulu-Natal and the eastern portions of the Free State and Eastern Cape. For more information on this Water Research Commission sponsored research, contact Dr Scott Sinclair (sinclaird@ukzn.ac.za) or visit our website for details and data _les (<http://sahg.ukzn.ac.za/>). We also welcome any feedback or comments you may have.

References:

- Pegram G.G.S., Sinclair S., Vischel T. and Nxumalo N., (2010), "Soil Moisture from Satellites: Daily maps over RSA for Flash Flood Forecasting, Drought Monitoring, Catchment Management and Agriculture", WRC Report No. 1683/1/10, Water Research Commission, Pretoria, South Africa.
- Sinclair S. and Pegram G.G.S., (2010), "A comparison of ASCAT and modelled Soil Moisture over South Africa, using TOPKAPI in land surface mode", Hydrol. Earth Syst. Sci., 14, 613-626.

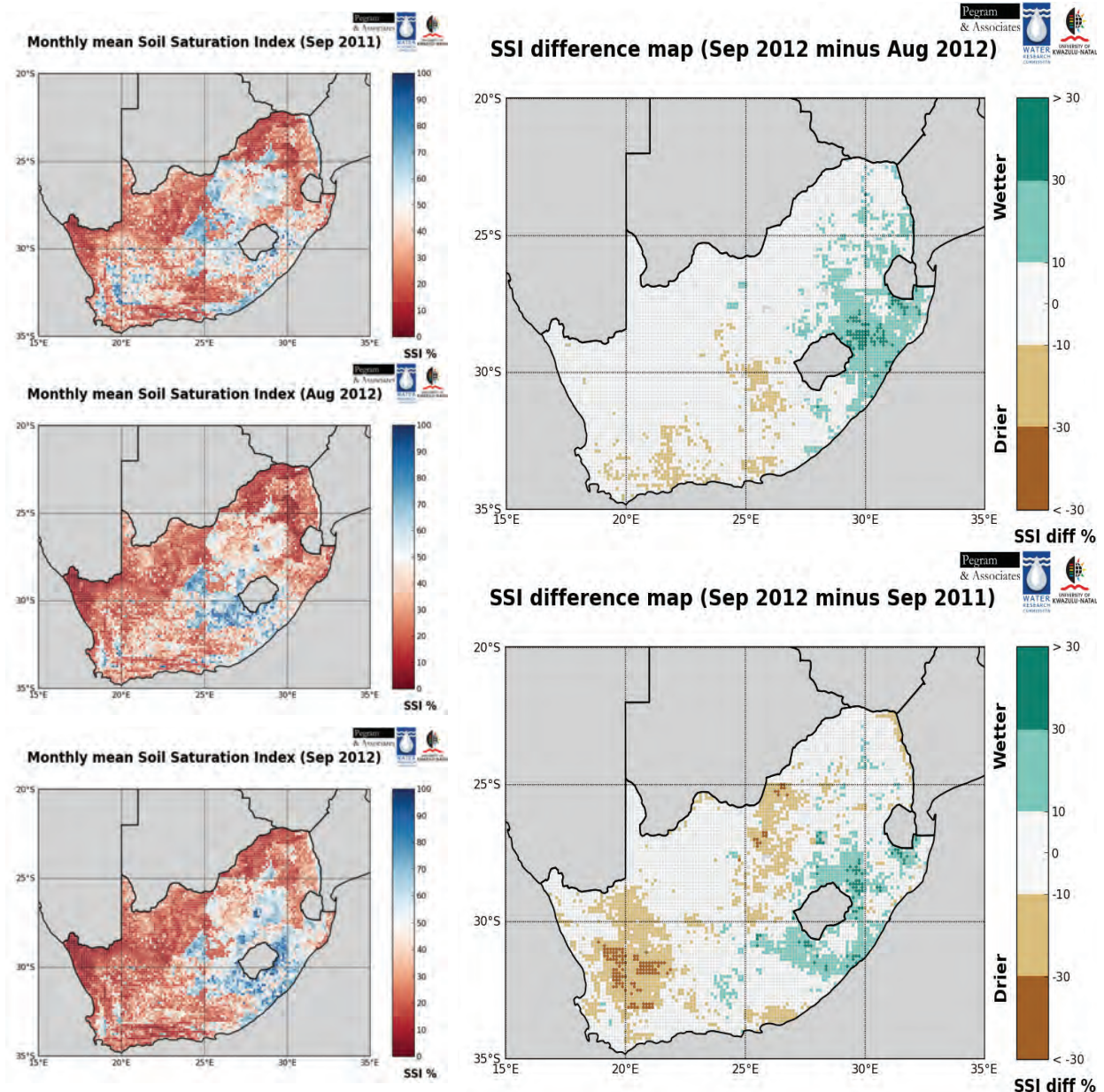


Figure 8.11. Typical images included monthly in Umlindi

8.3.2 Work done to get HYLARSMET running at ARC-ISCW.

We are keen to find an operational home for the HYLARSMET system, to ensure its continued usefulness after the conclusion of this project. The most promising outlook appears to be hosting the system at the ARC-ISCW; this was first suggested to us by Terry Newby and is currently being supported by Johan Malherbe of ARC-ISCW. The challenge has been to secure funding to make the transfer a reality. Through our participation in a WMO workshop held at the ARC-ISCW in Pretoria, we were fortunate enough to meet Robert Stefanski, Chief of the Agricultural Meteorology Division, Climate and Water Department, WMO who made an offer of S&T funding for Scott Sinclair to spend time in Pretoria setting up the system. It remains to find a source of funding for the computing hardware – some progress has also been made in this direction, as we discuss in section 4.

8.3.3 Collaborations with Roy Williams of ARC Onderstepoort veterinary institute.

We were contacted by Dr Roy Williams a Senior Researcher at the Onderstepoort Veterinary Institute. He is interested in using the HYLARSMET SSI and modelled overland flow volumes to forecast outbreaks of Rift-valley fever. After providing him with some of the HYLARSMET outputs, he was able to find a correlation between SSI and fever outbreaks. He is now involved in the discussions to get the system up and running at ARC-ISCW, and has even offered to assist with funding the computing hardware, since there is a benefit for his work.

8.3.4 Assisting model users

We have continued to assist users of the Open Source PyTOPKAPI model, at no charge for our services. Wherever possible these discussions are conducted on the public mailing list, which is archived at <http://groups.google.co.za/group/pytopkapi/>. Most recent assistance has given to the team working on WRC project K5-2162 “WRF Rainfall Parameterisation and Verification” – Michael Weston and Paulo Kagoda of e-Science.

An example transcript from the public mailing list follows:

On 29 November 2012 13:44, Mike_pytopkapi <westonf...@gmail.com> wrote:

```
> I have my tiffs for my new catchment. Everything looks good.  
> I am now trying to run ./create_file.py (version from the workshop)  
> I am getting the following error message (at end of mail) and really not  
> sure how to solve this.  
>  
> FYI, I am running create_file.py from the "generate_parameter_file" folder  
> that contains the tiffs.  
>  
> I added two lines to the end of create_file.py (file is attached)  
>  
> ini_fname = 'create-berg-param-file.ini'  
> generate_param_file(ini_fname)
```

It isn't necessary for you to run create_file.py directly. This file is part of the PyTOPKAPI library code and shouldn't be edited or moved from its install location. The idea is that you import the functions you need into your own python script and then run your script in your working directory to control the model (see example code below)

```
<code in e.g. my_pytopkapi_script.py>  
from pytopkapi.parameter_utils.create_file import generate_param_file
```

```
ini_fname = 'create-berg-param-file.ini'  
generate_param_file(ini_fname)  
</code in e.g. my_pytopkapi_script.py>
```

Run at the command line like so:

```
$ python my_pytopkapi_script.py
```

```
> Error message:  
>
```



```

> Traceback (most recent call last):
> File "./create_file.py", line 660, in <module>
>   generate_param_file(ini_fname)
> File "./create_file.py", line 214, in generate_param_file
>   cell_down)
> File "./create_file.py", line 486, in strahler_to_channel_manning
>   strahler_per_node[edge[0]] = stream_orders[key]
> KeyError: 0

```

The error is because your channel network and flow direction rasters combine to develop a disconnected network graph. I used the set of rasters you sent me offline to generate the Directed Graph of the channel network (berg-channel-network-graph.png attached), if you look at the second figure zoomed in on the upper part of the catchment, you'll notice many disconnected arcs. The numbers represent cells (nodes) in the network and the lines show the connections between nodes (heavy line on one end indicates direction). It's not possible to calculate the Strahler stream order from a discontinuous network, hence the failure. I'll see if there's a way to test for this and generate a more informative error message.

The channel network must be calculated using the flow direction raster. Most GIS tools do this by first computing a flow accumulation from the flow directions and then assigning channel cells based on a flow accumulation threshold. How did you create your channel network raster?

Cheers,
Scott

Attachments (2)

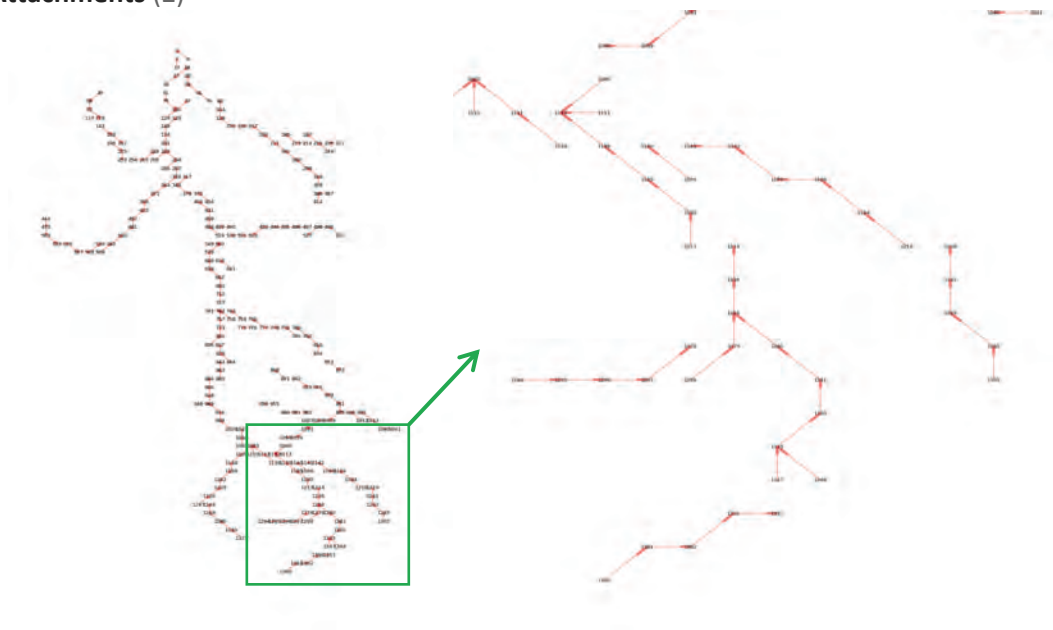


Figure 8.12. berg-channel-network-graph berg-channel-network-graph-zoom

8.3.5 Collaboration with Emanuel Dutra and Florian Pappenberger of the ECMWF

These two scientists work for the European Centre for Medium-range Weather Forecasting (ECMWF), are colleagues on the GLOWASIS team and are interested in comparing the ECMWF ERA interim Soil Moisture estimates with our HYLARSMET SSI. An example of the type of comparisons to be made between our work and theirs follows.

Regression (PyTOPKAPI/PCRGLOBWB-ERA)

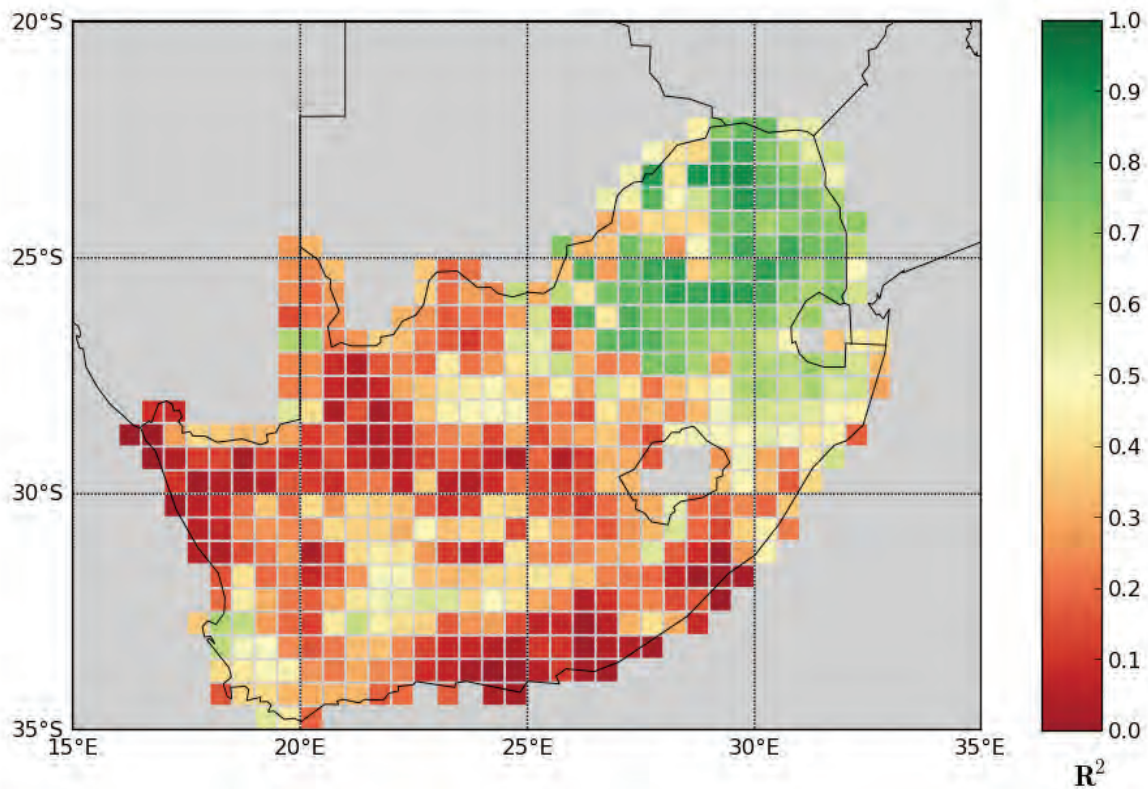


Figure 8.13 Map of R^2 computed for each 0.5° grid cell of HYLARSMET versus PCR-GLOBWB ERA. Grey areas due to no data from one (or both) of the data-sets during the study period, or due to at least one of the data-sets having a value of zero throughout the period.

8.3.6 GLOWASIS, Utrecht & Innovation Hub, Pretoria

Three presentations were made on our work at the GLOWASIS meetings in Holland in February and October 2011 and in October 2012. Dr Sinclair also made a presentation at the Conference on Space Partnerships for Sustainable Development in April 2011 at the Innovation Hub in Pretoria. This conference was attended by high-level delegates from the Department of Science and Technology and their EU counterparts.

8.3.7 Publications

2 papers on PyTOPKAPI during the project:

Sinclair S. and Pegram G.G.S. (2010). A comparison of ASCAT and modelled Soil Moisture over South Africa, using TOPKAPI in land surface mode, *Hydrol. Earth Syst. Sci.*, 14, 613-626.

Sinclair S., Pegram G.G.S. (2012). A sensitivity assessment of the TOPKAPI model with an added infiltration module, *Journal of Hydrology*, Available online 7 December 2012, doi:<http://dx.doi.org/10.1016/j.jhydrol.2012.11.061>

Two presentations at European Geophysical Union Congress on PyTOPKAPI and HYLARSMET.

Sinclair, Scott and Geoff Pegram (2011). PyTOPKAPI – an open source hydrological model used to estimate Soil Moisture in near real time over South Africa for flood potential & drought monitoring and remote sensing model intercomparison, European Geosciences Union general Assembly, Vienna

Sinclair DS and GGS Pegram (2010). A comparison of ASCAT and modelled Soil Moisture over South Africa, using TOPKAPI in land surface mode, European Geosciences Union general Assembly, Vienna

Two presentations at SANCIAHS 2012 on PyTOPKAPI and HYLARSMET.

Sinclair S. and G.G.S. Pegram (2012). PyTOPKAPI – an Open-Source Implementation of the TOPKAPI Hydrological Model, 16th SANCIAHS SYMPOSIUM. Pretoria, 1-3 October

Sinclair S. and G.G.S. Pegram (2012) Modelling Soil Moisture at National and Catchment Scale using a Spatially Distributed Hydrological Model forced by Remote Sensing and Meteorological Products , 16th SANCIAHS SYMPOSIUM. Pretoria, 1-3 October

8.3.8 Lectures/presentations given at WATERNET MSc course hosted by BEEH in September 2012

Dr Sinclair spent two days lecturing a group of four SADC MSc students in Water Resources based in Pietermaritzburg (morning lectures were followed by afternoon practical sessions using a virtual machine on USB stick so that each student had an identical software environment to work in). Lectures were on rainfall and Soil Moisture modelling and included content from our HYLARSMET work. The lectures are available in pdf form from the dropbox files which follow, because they are large [8Mb and 28Mb respectively]:

<http://dl.dropbox.com/u/23555555/sinclair-waternet-rainfall.pdf>

<http://dl.dropbox.com/u/23555555/sinclair-waternet-soil-moisture.pdf>

The Practical session notebooks (solutions) are available at:

<http://nbviewer.ipython.org/4344350/>

<http://nbviewer.ipython.org/4344276/>

<http://nbviewer.ipython.org/4344376/>

Some selected slides from the rainfall presentation follow:

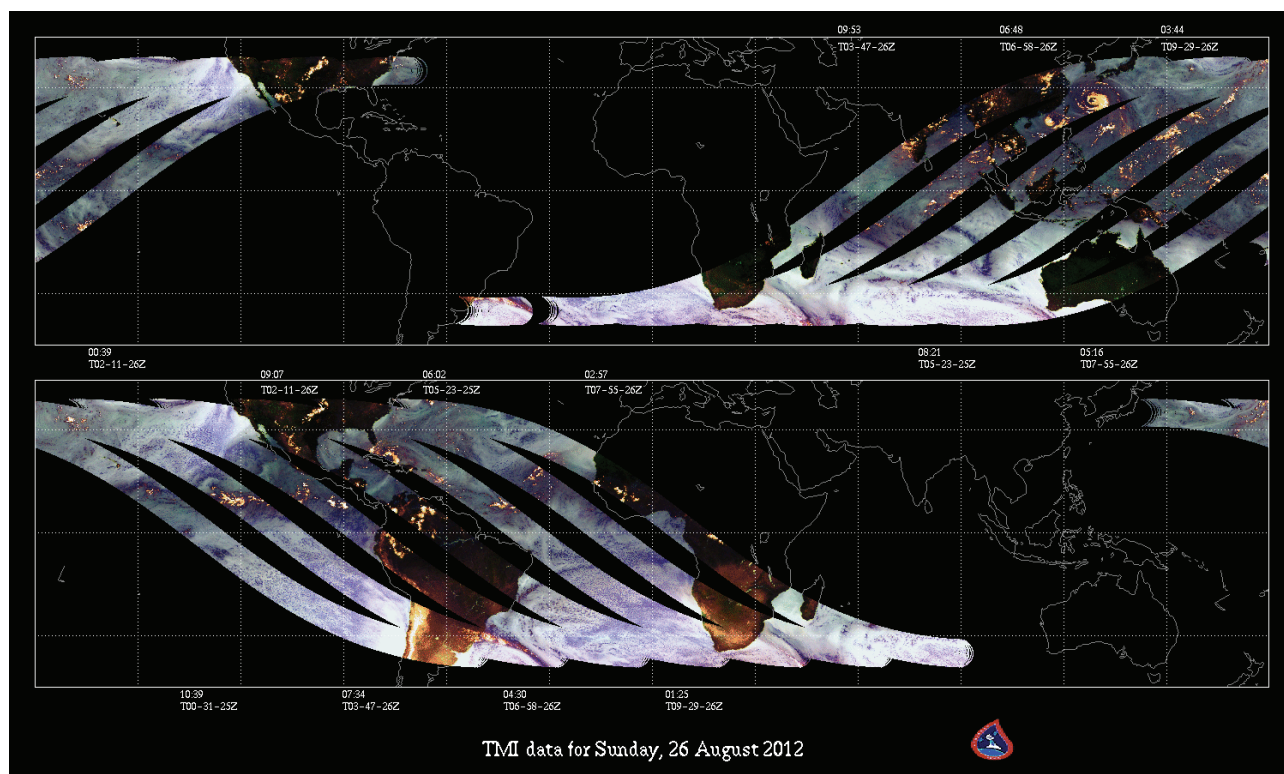


Figure 8.14 TRMM swath on one day: 26 August 2012.

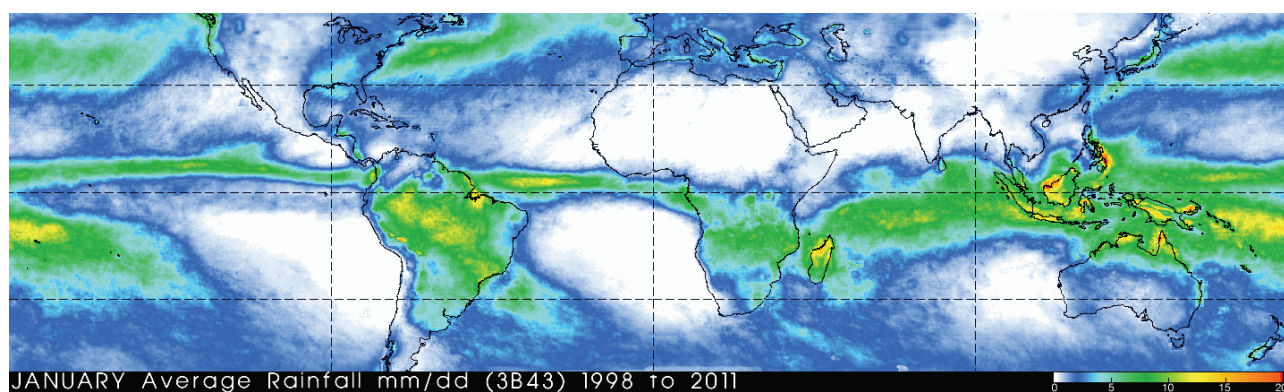


Figure 8.15. Global TRMM January Average rainfall

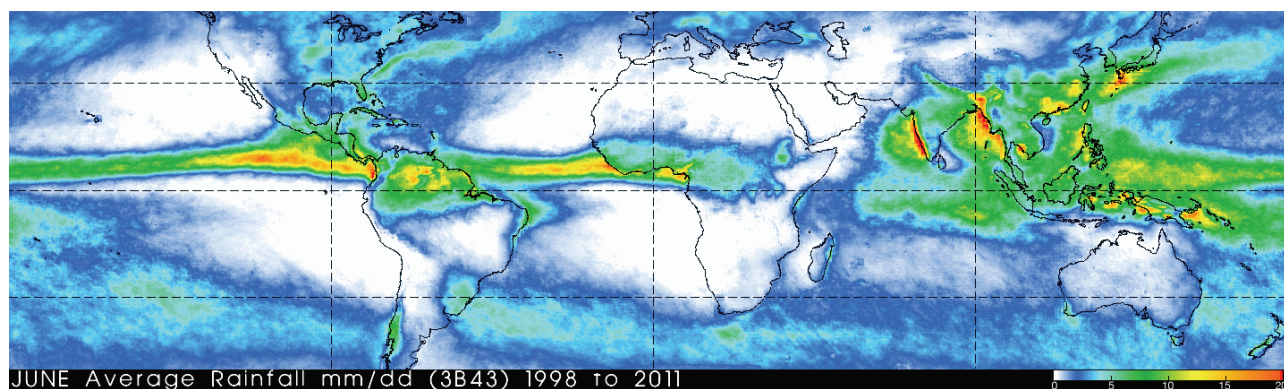


Figure 8.16. Global TRMM June Average rainfall – note the Northward shift over mid-Africa

From the Soil Water presentation:

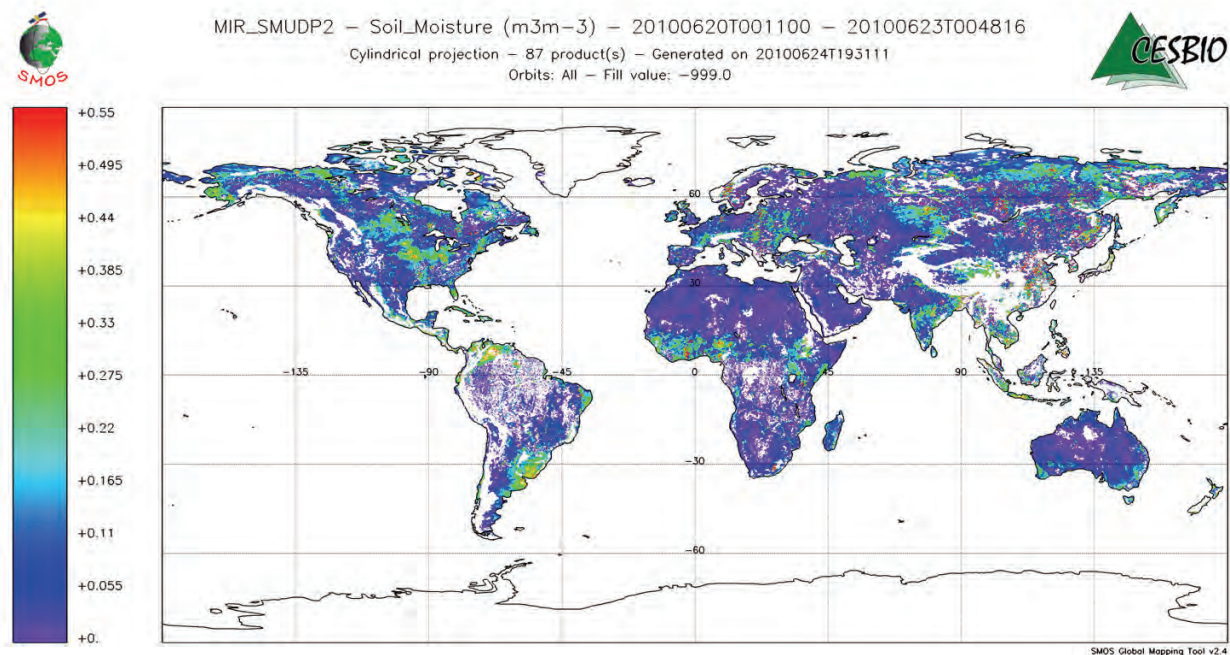


Figure 8.17. SMOS estimate of Global Soil Moisture : 20 to 23 June 2010

The lectures were designed to share the richness of information which is now becoming routinely available to researchers.

The following material is a sample of the workshop problem solving session material designed and offered by Dr Sinclair.

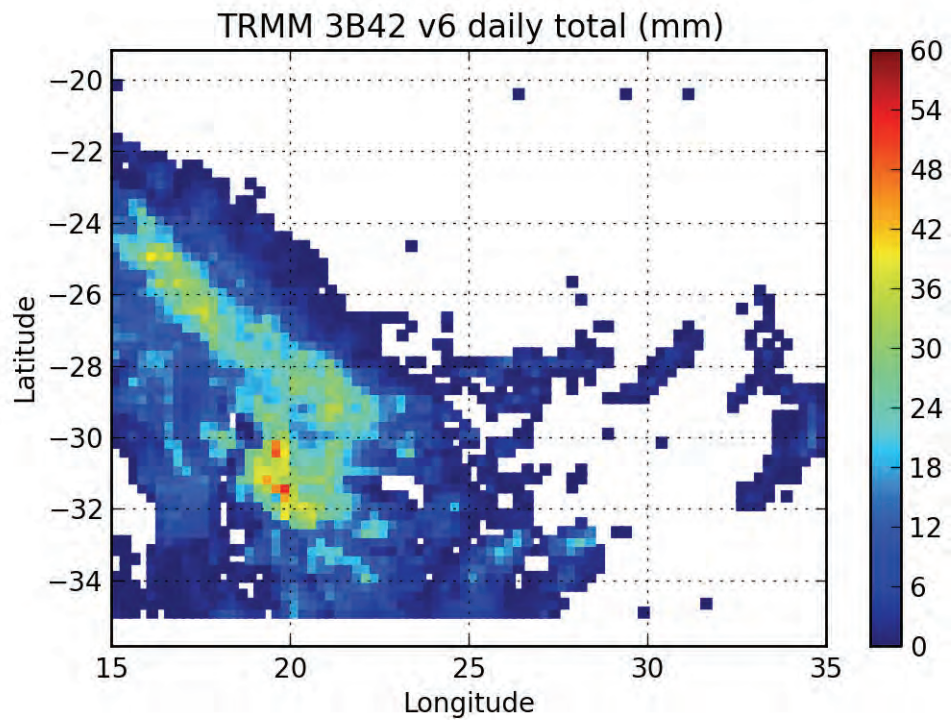
Influence of gauge correction on rainfall estimates

Use the techniques you learned in the earlier notebook to make a comparison between the gauge adjusted 3B42 (the netCDF files in the data/3B42 directory) and the realtime product (binary files in the data/3B42RT directory). You already know how to handle the NetCDF files for the gauge adjusted rainfall estimates, so you'll need to learn how to handle the real-time binary files. Luckily you've been provided with PyTRMM a handy Python library to read the data contained in the TRMM binary file format. An example of how to use the PyTRMM library is given below, use the lat and lon arrays you get from the NetCDF files to help you plot the precip field from the binary file.

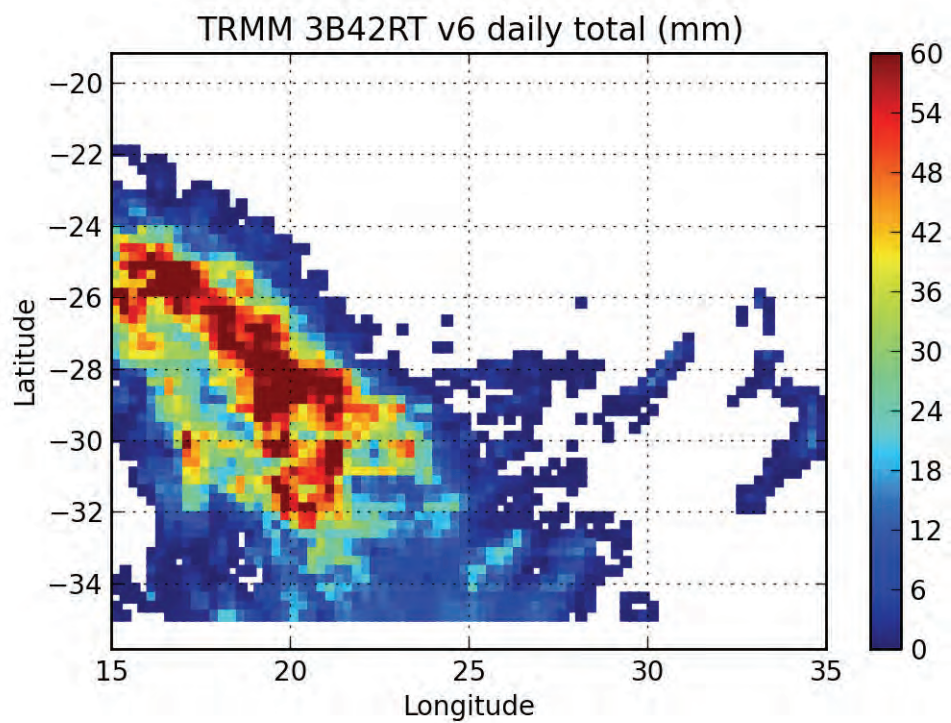
- Make sure that you are comparing algorithm version 6 for gauge adjusted and real-time versions
- Try to write two Python functions that read each kind of file and return the lat, lon and precip arrays

To get the following images.

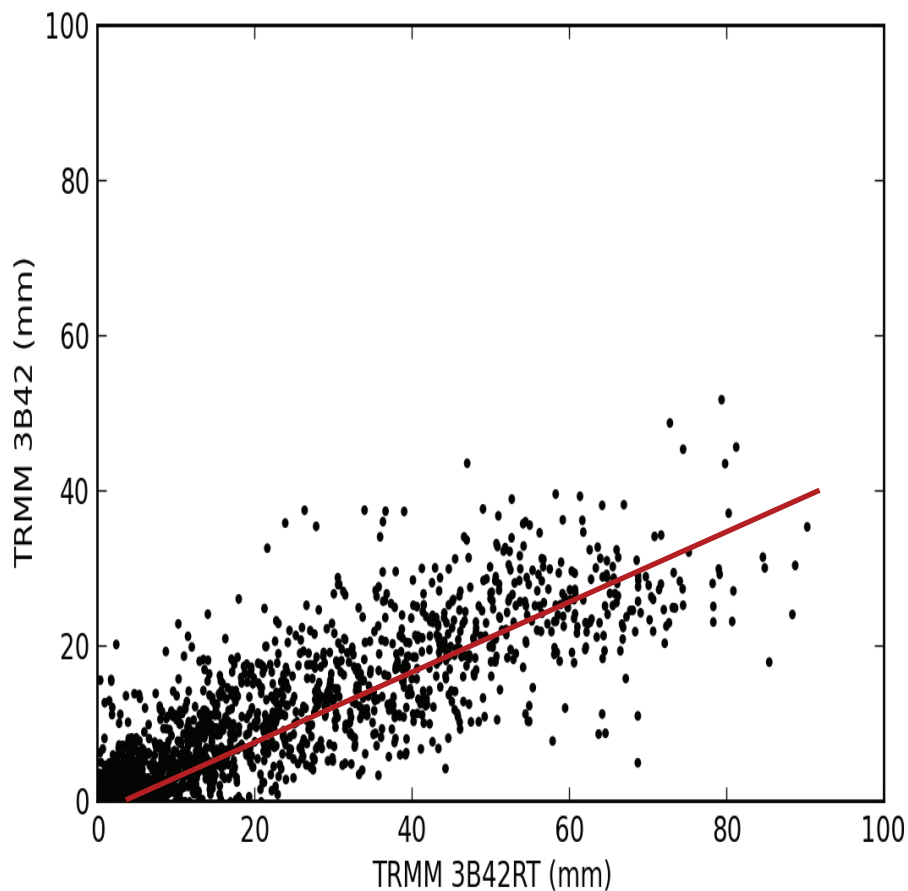
First the corrected TRMM:



Now the Real Time estimate:



Now make a pixel-by-pixel comparison:



What do the results tell you?

Self study – Comment on the relationship between the two algorithm versions...

Bonus credit – Fit a regression line to the scatter plot of 3B42v7 versus 3B42v6

-----///-----

8.4 Summary

The outreach and sharing of our research has been on a variety of fronts, from personal contacts in the flesh and on the internet, through small group discussions to presentations and workshops through to presentations at National and International Symposia and Congresses, then by dissemination through papers in International Journals.

We have formed strong links with local and International Scientists, Engineers and Decision Makers, who have not only listened to our story, but have invited us to participate meaningfully in a range of research activities.

We believe that we have made the proud name of the WRC well known and honoured for its continued support of top class research. These and similar activities will continue throughout the remaining time of this project and, most likely, some way beyond that.

9. Conclusion

This report summarises the work we have done on the WRC project HYLARSMET: A HYdrologically consistent LAnd Surface Model for Soil Moisture and EvapoTranspiration modelling over Southern Africa using Remote Sensing and METeorological data.

We have:

- Obtained and used updated soil and land use data from ARC-ISCW and, while working on this, realised that the WRC funded Agrohydrological atlas of Schulze et al. (2008) provided the missing soil properties derived from the ISCW data and so we elected to use this information to infill the missing regions and simultaneously update the soil properties for the whole country.
- Set up source code hosting, bug tracking, mailing list and website for HYLARSMET to enable collaborators to more easily use and contribute to the development of the model that forms the core of the project's work.
- Added a Green-Ampt surface infiltration layer to TOPKAPI formulation and tested its efficacy and, what is more, tested the sensitivity of the PyTOPKAPI model to changes in the Soil parameters and Rainfall; we found that the Rainfall dominates the ET as far as sensitivity goes.
- Installed hydrologically consistent implementations of HYLARSMET on several SAWS FFG target catchments and computed Soil Moisture at fine scale; in the process we automated the setting up a PyTOPKAPI model in new catchments, so that the fine scale modelling of many SAFFG catchments becomes feasible.
- Back calculating SWI estimates to 1 August 2008, using the best available parameters and forcing data from years 1 and 2. In addition, we exceeded this original intention by first doing the back-calculations on a monthly basis for the ARC-ISCW Umlindi newsletter in an ad-hoc way by hand, and then developing a tool-chain to automate this process in an easy and repeatable manner. Throughout the time that we have been producing monthly Soil Moisture assessments we have been archiving the source data-sets and building on the work done in previous work in the project to improve the HYLARSMET SSI modelling procedure.
- Compared modelled Soil Moisture to a suitable subset of readily available remote sensing Soil Moisture estimates making comparisons of the PyTOPKAPI Soil Moisture estimates against the products of a two-year European Union FP7 project called GLOWASIS [Global Water Scarcity Information Service] in which we have been partners.
- Maintained our links with the HYLARSMET community by fixing programming bugs, extending the model, maintaining the UKZN website which hosts our recent achievements, encouraged user interaction, offered workshops, published in top journals and presented our findings in South Africa and internationally at conferences.

We believe that we have had a good three years of establishing our product as practical and useful, not just as a successful academic research exercise. Where to from here? There are several areas of

hydrometeorology which would benefit from extending the capabilities of the HYLARSMET and PyTOPKAPI formulations. The first is to extend the time period of analysis from the 4 ½ years we have done so far [August 2008 to February 2013]; we could certainly back-calculate to 2000, or possibly earlier, once we obtain the earlier forcing rainfall and Met dat. This extension would be valuable for giving a baseline of records to allow the determination of anomalies and changes. Then there is the need to spatially intensify the model to cover more than 1 sq km in every 160 sq km, in order to provide spatial continuity and detail of Soil Moisture and actual evaporation over every square kilometre of the whole country. This would immediately combine HYLARSMET and PyTOPKAPI into one model which would valuably provide near real time runoff information over the whole of RSA, in detail – this would assist in improving and informing flood forecasting, agriculture (e.g. Dry land wheat farming in the Free State) and water resources analysis and planning. Looking further afield, we need to share this bounty with our SADC neighbours; we have shown in this study that this can be achieved with some confidence, because the African soil maps are available and TRMM near real time rainfall is global. With a new home for the computation and data store in Agricultural Research Council's Institute for Soil Climate and Water, this powerful product we have developed will not be lost, but can endure as long as it is useful – the longer the better!

References

- 1 Allen R.G., Pereira L.S., Raes D., Smith M. 1998. Crop evapotranspiration – Guidelines for computing crop water requirements. FAO Irrigation and drainage paper 56. Rome.
- 2 Brooks, R. and Corey, A. (1964). Hydraulic properties of porous media. Technical report, Hydrology Paper No. 3, *Colorado State University*, Fort Collins, Colorado.
- 3 Chow, V., Maidment, D., and Mays, L. (1988). *Applied Hydrology*. McGraw-Hill.
- 4 De Lange R., Beck R., Van De Giesen N., Friesen J., De Wit A. and Wagner W. (2008). Scatterometer-Derived Soil Moisture Calibrated for Soil Texture With a One-Dimensional Water-Flow Model. *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 46, No. 12, 4041-4049.
- 5 De Roo, A.P.J., Wesseling, C.G. and Van Deursen, W.P.A. (2000) Physically based river basin modelling within a GIS: The LISFLOOD model. *Hydrological Processes*, 14, pp. 1981–1992.
- 6 El-Kadi, A. (1985). On estimating the hydraulic properties of soils, part 1. comparison between forms to estimate the soil-water characteristic function. *Adv. Water Res.*, 8:136-147.
- 7 GLCC, “Global Land Cover Characteristics”, United States Geological Survey (USGS), <http://edcdaac.usgs.gov/glcc>, 1997.
- 8 Gleyzer A., Denisyuk M., Rimmer A. and Salingar Y., 2004. A Fast Recursive GIS Algorithm for Computing Strahler Stream Order in Braided and Non-braided Networks. *Journal of the American Water Resources Association (JAWRA)* 40(4):937-946.
- 9 Green, W. and Ampt, G. (1911). Studies on soil physics, part 1, the flow of air and water through soils. *J. Agric. Sci.*, 4(1):1-24.
- 10 Huffman, G., Adler, R., Bolvin, D., Gu, G., Nelkin, E., Bowman, K., Hong, Y., Stocker, E., and Wolff, D. (2007). The TRMM multi-satellite precipitation analysis: Quasi- global, multi-year, combined-sensor precipitation estimates at fine scale. *J. Hydrometeor.*, 8(1):38-55.
- 11 Huffman, G. (2012). *pers. comm.*
- 12 Jarvis A., Reuter H.I., Nelson A., Guevara E., “Hole-filled seamless SRTM data V4”, International Centre for Tropical Agriculture (CIAT), available from <http://srtm.csi.cgiar.org>, 2008.
- 13 Jones, E., Oliphant, T., Peterson, P., et al. (2001). SciPy: Open source scientific tools for Python.
- 14 Liu, Y.Y., Parinussa R.M. , Dorigo W.A., De Jeu R.A.M., Wagner W., van Dijk A.I.J.M., McCabe M.F., and Evans J.P. (2011). Developing an improved Soil Moisture dataset by blending passive and active microwave satellite-based retrievals. *Hydrol. Earth Syst. Sci.*, 15, 425–436.
- 15 Liu, Z. and Todini, E. (2002). Towards a comprehensive physically-based rainfall-runoff model. *Hydrol. Earth Syst. Sci.*, 6(5):859-881.
- 16 Ma, Y., Feng, S., Su, D., Gao, G., and Huo, Z. (2010). Modeling water infiltration in a large layered soil column with a modified Green-Ampt model and HYDRUS-1D. *Computers and Electronics in Agriculture*, 71:S40-S47.
- 17 McNeill, L., Brandão, A., Zucchini, W., Joubert, A. (1994). Interpolation of the daily rainfall model. *WRC Report No. 305/1/94*. Water Research Commission: Pretoria.
- 18 Middleton, B. and Bailey, A. (2009). Water resources of South Africa, 2005 study. Technical report, Water Research Commission Report No. TT 380/08, *Water Research Commission*, Pretoria, South Africa.

- 19 Moré, J., Garbow, B., and Hillstrom, K. (1980). User guide for MINPACK-1. Technical report, *Argonne National Laboratory Report ANL-80-74*, Argonne, Ill.
- 20 Pegram GGS, DS Sinclair, Parak M, Sakulski D, and Nxumalo N, (2007) National Flood Nowcasting System: Towards an Integrated Mitigation Strategy, Water Research Commission Report WRC 1429/1/06; ISBN 978-1-77005-512-1
- 21 Pegram, G., Sinclair, S., Vischel, T., and Nxumalo, N. (2010). Soil Moisture from satellites: Daily maps over RSA for flash flood forecasting, drought monitoring, catchment management & agriculture. Technical Report, Water Research Commission Report No. K5-1683, *Water Research Commission*, Pretoria, South Africa.
- 22 Powell, M. (1970). Numerical Methods for Nonlinear Algebraic Equations, chapter A *Hybrid Method for Nonlinear Equations*. Gordon and Breach.
- 23 Rawls, W., Brakensiek, D., and Saxton, K. (1982). Estimation of soil water properties. *Trans. of the ASAE*, 25(5):1316-1320 and 1328.
- 24 Rouault, M., Fauchereau, N., Pohl, B., Penven, P., Richard, Y., Reason, C.J.R., Pegram, G.G.S., Phillippon, N., Siedler, G. and Murgia, A. (2010). Multidisciplinary Analysis of Hydroclimatic Variability at the Catchment Scale. *WRC Report No. 1747/1/10*, Water Research Commission: Pretoria.
- 25 Schulze, R., Maharaj, M., Warburton, M., Gers, C., Horan, M., Kunz, R., and Clark, D. (2008). South African atlas of climatology and Agrohydrology. Technical report, Water Research Commission Report No. 1489/1/08, *Water Research Commission*, Pretoria, South Africa.
- 26 Sinclair S. and Pegram G.G.S. (2010). A comparison of ASCAT and modelled Soil Moisture over South Africa, using TOPKAPI in land surface mode, *Hydrol. Earth Syst. Sci.*, 14, 613-626.
- 27 Sinclair DS and GGS Pegram (2010). A comparison of ASCAT and modelled Soil Moisture over South Africa, using TOPKAPI in land surface mode, European Geosciences Union general Assembly, Vienna
- 28 Sinclair, Scott and Geoff Pegram (2011). PyTOPKAPI – an open source hydrological model used to estimate Soil Moisture in near real time over South Africa for flood potential & drought monitoring and remote sensing model intercomparison, European Geosciences Union general Assembly, Vienna
- 29 Sinclair S. and G.G.S. Pegram (2012a) Modelling Soil Moisture at National and Catchment Scale using a Spatially Distributed Hydrological Model forced by Remote Sensing and Meteorological Products , 16th SANCIAHS SYMPOSIUM. Pretoria, 1-3 October
- 30 Sinclair S. and G.G.S. Pegram (2012b). PyTOPKAPI – an Open-Source Implementation of the TOPKAPI Hydrological Model, 16th SANCIAHS SYMPOSIUM. Pretoria, 1-3 October
- 31 Sinclair S., Pegram G.G.S. (2013). A sensitivity assessment of the TOPKAPI model with an added infiltration module, *Journal of Hydrology*, Vol 478, pp 100-112
doi:<http://dx.doi.org/10.1016/j.jhydrol.2012.11.061>
- 32 SIRI, “Land Type Series”, Department of Agriculture and Water Supply, Soil and Irrigation Research Institute, Memoirs on the Agricultural Natural Resources of South Africa, Pretoria, 1987.
- 33 Stocker E. (2012). *pers. comm.*
- 34 Van Beek, L.P.H. and M.F.P. Bierkens (2008), The Global Hydrological Model PCR-GLOBWB: Conceptualization, Parameterization and Verification, *Report Department of Physical Geography*, Utrecht University, Utrecht, The Netherlands,
<http://vanbeek.geo.uu.nl/supinfo/vanbeekbierkens2009.pdf>

- 35 Van der Knijff J.M., J. Younis and A.P.J. De Roo (2010) LISFLOOD: a GIS-based distributed model for river-basin scale water balance and flood simulation. *International Journal of Geographical Information Science*, Vol. 24, No.2, 189-212.
- 36 Vischel, T., Pegram, G., Sinclair, S., and Parak, M. (2008a). Implementation of the TOPKAPI model in South Africa: Initial results from the Liebenbergsvlei catchment. *Water SA*, 34(3):1-12.
- 37 Vischel, T., Pegram, G., Sinclair, S., Wagner, W., and Bartsch, A. (2008b). Comparison of Soil Moisture _elds estimated by catchment modelling and remote sensing: a case study in South Africa. *Hydrol. Earth Syst. Sci.*, 12:1-17.
- 38 Zucchini, W. and Adamson, P.T. (1984). The occurrence and severity of droughts in South Africa. *WRC Report No. 91/1/84*, Water Research Commission: Pretoria.

APPENDIX



HYLARSMET
WRC PROJECT K5/2024

1st PyTOPKAPI workshop

Prof. Geoff PEGRAM

Dr. Scott SINCLAIR



13 February 2012

The purpose of this document is to provide a tutorial introduction on the use of PyTOPKAPI, a BSD licensed Python library implementing the TOPKAPI Hydrological model (Liu and Todini, 2002).

The tutorial session is meant to be hands-on and it's expected that the workshop participants follow all of the examples on their own computers. Each participant has been provided with a bootable USB flash drive containing a customized live install of Ubuntu Linux. This will ensure that everyone has an identical computing environment during the tutorial.

1 Introduction to the working environment

In the first part of the tutorial, we'll develop some basic familiarity with the customized Ubuntu Linux environment that you'll be using to run the PyTOPKAPI model. The model can also run on many other computer operating systems including Microsoft Windows and Apple Mac. Our primary method of interaction will be via the command line as this provides a powerful and repeatable way of generating model results, that can also be scripted to avoid manually performing the same procedures when multiple model runs are required.



Fig. 1: The Ubuntu Linux desktop as it appears after booting the computer using the live USB. The application launcher appears on the left side of the screen.

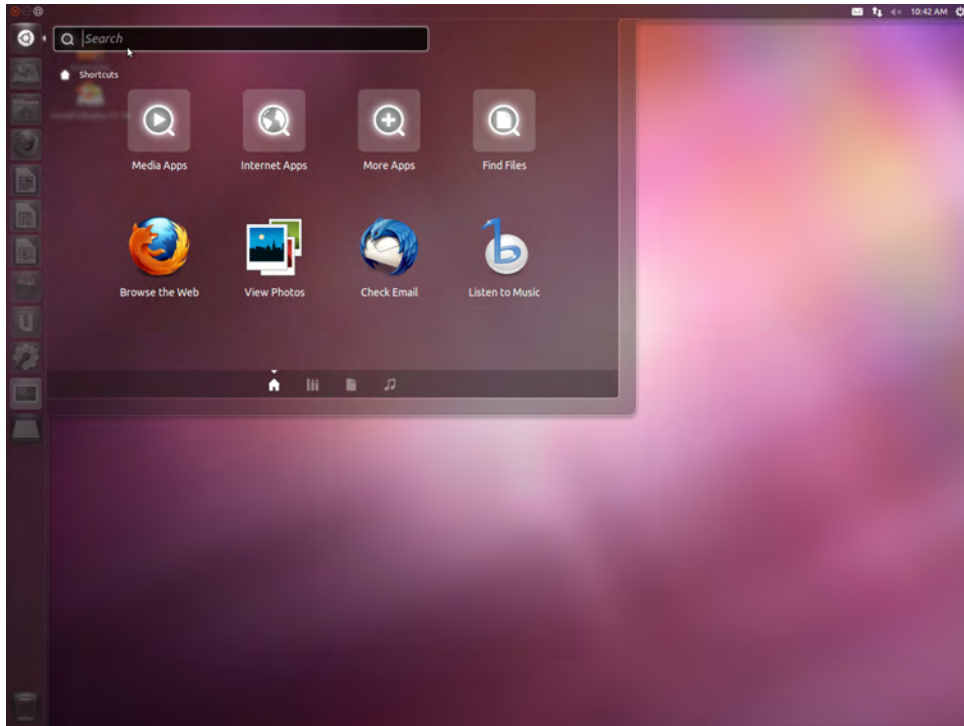


Fig. 2: Clicking on the icon at the top of the launcher opens the system dashboard, which can be used to search for applications.

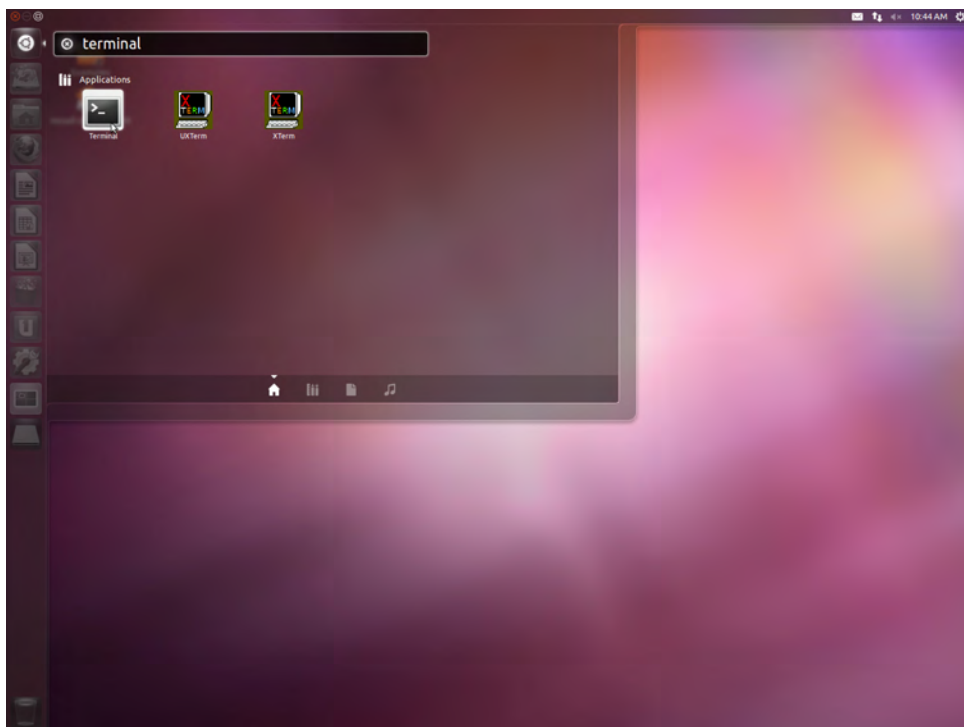
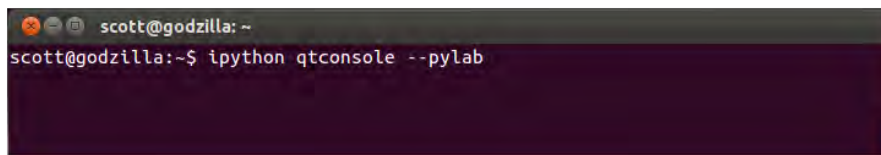
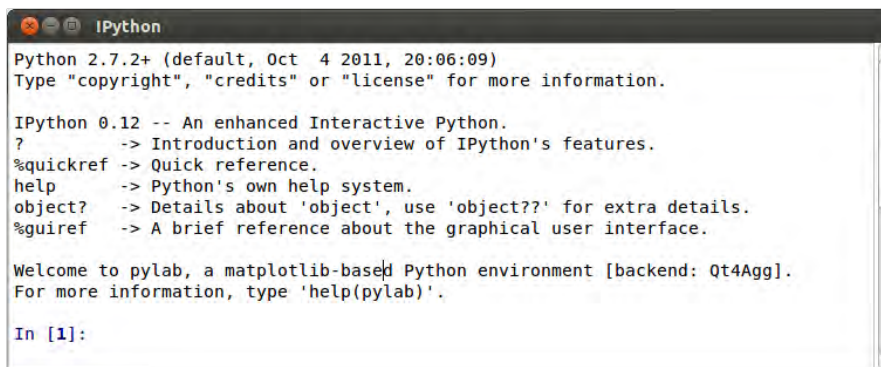


Fig. 3: Typing *terminal* into the search bar shows a selection of terminal applications installed on the system. Click on the Terminal icon to start a Terminal instance.

A terminal window with a dark background. The prompt is 'scott@godzilla: ~'. The command 'ipython qtconsole --pylab' has been entered and executed. The terminal is currently empty below the command.

```
scott@godzilla: ~  
scott@godzilla:~$ ipython qtconsole --pylab
```

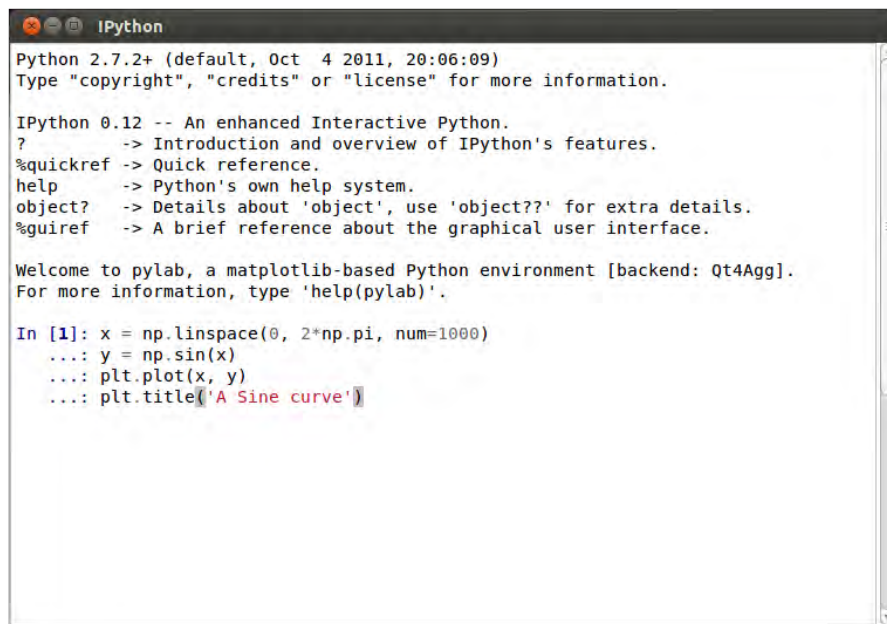
Fig. 4: We are going to launch an interactive Python environment called IPython from the terminal. You'll see the advantages of IPython later on, for now it isn't necessary to understand the command entered in the terminal, it's sufficient to type it in and press *Enter* on your keyboard.

A window titled 'IPython'. It displays the IPython version (0.12) and provides a list of help commands: '?', '%quickref', 'help', 'object?', and '%gui'. It also includes a welcome message for 'pylab' and a prompt 'In [1]:'.

```
IPython  
Python 2.7.2+ (default, Oct  4 2011, 20:06:09)  
Type "copyright", "credits" or "license" for more information.  
  
IPython 0.12 -- An enhanced Interactive Python.  
?          -> Introduction and overview of IPython's features.  
%quickref  -> Quick reference.  
help       -> Python's own help system.  
object?    -> Details about 'object', use 'object??' for extra details.  
%gui       -> A brief reference about the graphical user interface.  
  
Welcome to pylab, a matplotlib-based Python environment [backend: Qt4Agg].  
For more information, type 'help(pylab)'.  
  
In [1]:
```

Fig. 5: The IPython console application. Once this window has appeared, you can minimize the terminal window to clear screen clutter. We'll develop familiarity with the the scientific python workflow using IPython in the next section.

Basic use of Numpy and Matplotlib in IPython



```
Python 2.7.2+ (default, Oct 4 2011, 20:06:09)
Type "copyright", "credits" or "license" for more information.

IPython 0.12 -- An enhanced Interactive Python.
?          -> Introduction and overview of IPython's features.
%quickref  -> Quick reference.
help       -> Python's own help system.
object?    -> Details about 'object', use 'object??' for extra details.
%gui?      -> A brief reference about the graphical user interface.

Welcome to pylab, a matplotlib-based Python environment [backend: Qt4Agg].
For more information, type 'help(pylab)'.

In [1]: x = np.linspace(0, 2*np.pi, num=1000)
...: y = np.sin(x)
...: plt.plot(x, y)
...: plt.title('A Sine curve')
```

Fig. 6: A sequence of commands in IPython to produce a plot of a Sine curve. You should type these commands exactly as they appear, using *Ctrl-Enter* to move to the next line without executing the command. We'll discuss what these commands mean a little later. For now just execute them by pressing *Enter* twice after entering the final command. A plot window will appear.

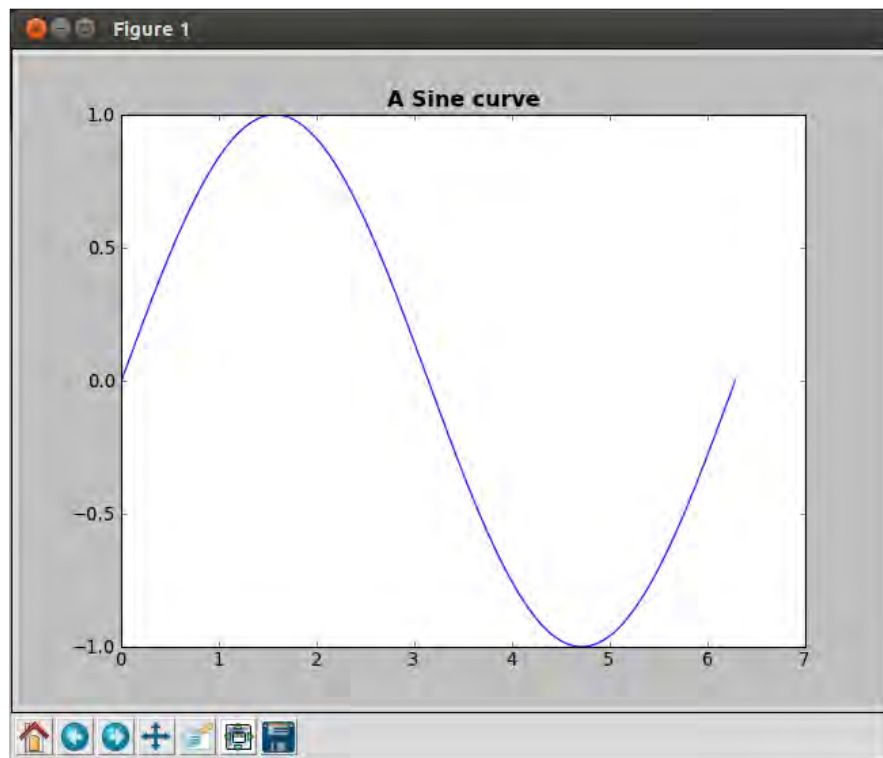


Fig. 7: The plot window showing our basic Sine curve plot. The plot can easily be further customized as we'll see during the workshop session.

2 Example model simulation

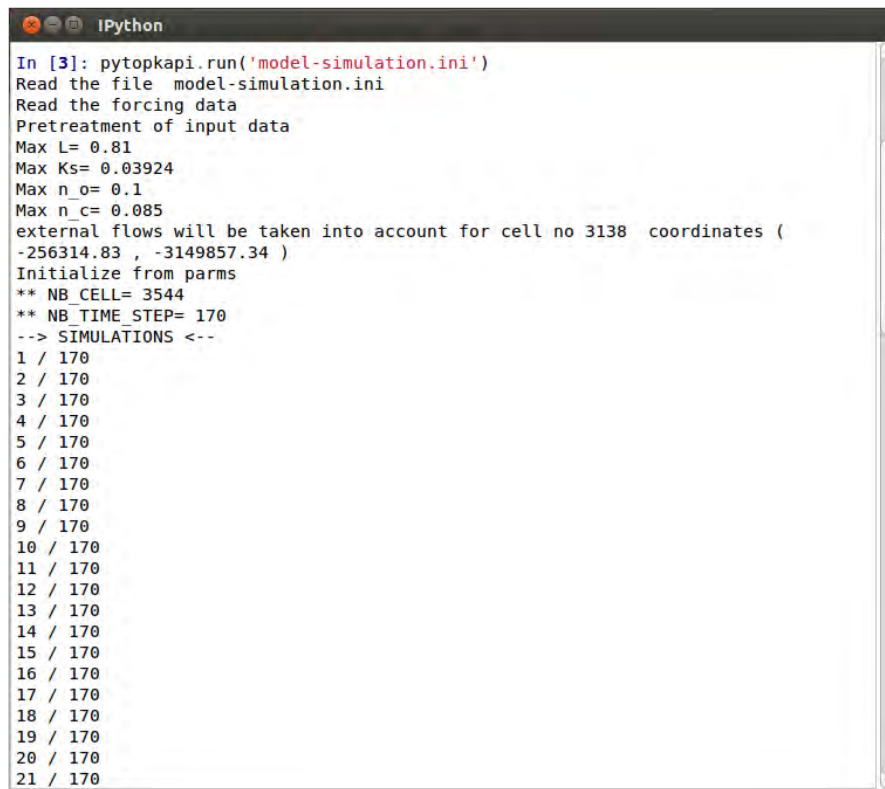
As a first introduction to the model, we'll run a complete pre-packaged example simulation on the Liebenbergsvlei catchment and analyse the simulation results using the tools introduced in section 1.

Enter the following sequence of commands in your IPython window (pressing *Enter* after each command to execute the instruction):

```
cd ~/PyTOPKAPI_workshop/example_simulation

import pytopkapi
pytopkapi.run('model-simulation.ini')
```

You should see the following:



```
IPython
In [3]: pytopkapi.run('model-simulation.ini')
Read the file model-simulation.ini
Read the forcing data
Pretreatment of input data
Max L= 0.81
Max Ks= 0.03924
Max n_o= 0.1
Max n_c= 0.085
external flows will be taken into account for cell no 3138 coordinates (
-256314.83 , -3149857.34 )
Initialize from parms
** NB_CELL= 3544
** NB_TIME_STEP= 170
--> SIMULATIONS <--
1 / 170
2 / 170
3 / 170
4 / 170
5 / 170
6 / 170
7 / 170
8 / 170
9 / 170
10 / 170
11 / 170
12 / 170
13 / 170
14 / 170
15 / 170
16 / 170
17 / 170
18 / 170
19 / 170
20 / 170
21 / 170
```

While the model simulation is running, we'll discuss the meaning of the commands you've just issued.

```
cd ~/PyTOPKAPI_workshop/example_simulation
```

The `cd` command changes your current working directory to one that contains the required input files for this model simulation. In Linux, directories are separated by a forward slash `/` not a backward slash `\` as in Windows. The `~` symbol is a shortcut for the path to your home directory.

```
import pytopkapi
```

This command imports the `pytopkapi` Python package so that the model and its tools can be used by calling commands in the IPython session.

```
pytopkapi.run('model-simulation.ini')
```

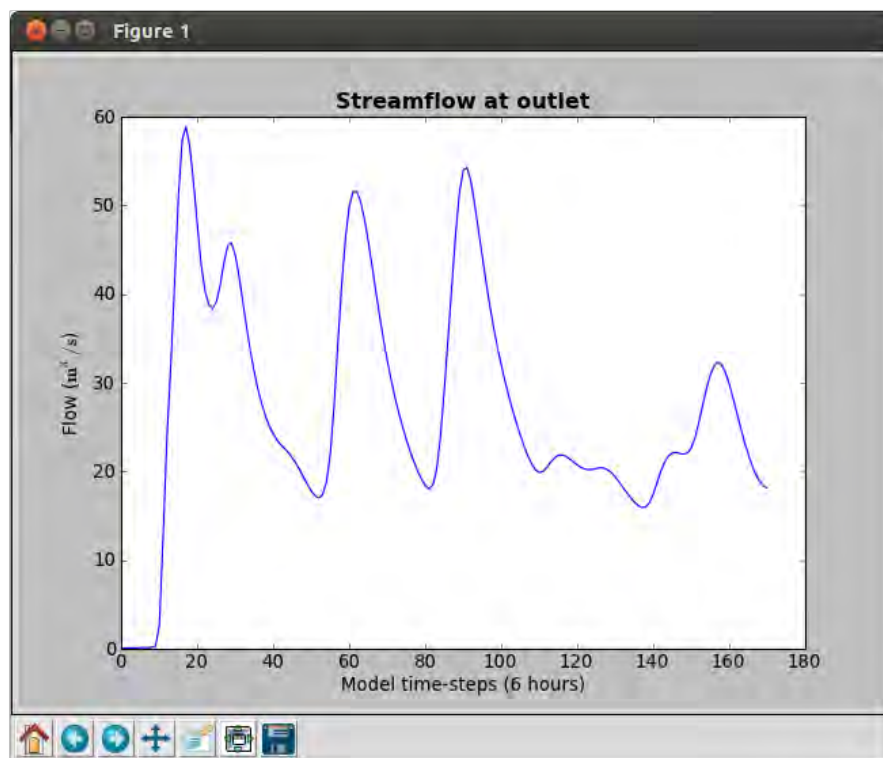
Runs the model using the parameter and forcing files specified in the configuration file *model-simulation.ini*.

Once the model simulation has completed, there should be a file called *Example_simulation_results.h5* in the *results* sub-directory. This file is in a file format called HDF5, which is well suited to the efficient storage and query of large numbers of data points. We won't discuss the format of this file but will discover how to extract interesting data from it.

Our next task will be to open the simulation file and plot some of the results. To obtain and plot the modelled streamflow at the catchment outlet, enter the code below into the IPython console:

```
import h5py
h5file = h5py.File('results/Example_simulation_results.h5')
channel_flows = h5file['Channel/Qc_out'][...]
plt.plot(channel_flows[:, 0])
plt.title('Streamflow at outlet', fontweight='bold')
plt.ylabel('Flow ( $\text{m}^3/\text{s}$ )')
plt.xlabel('Model time-steps (6 hours)')
```

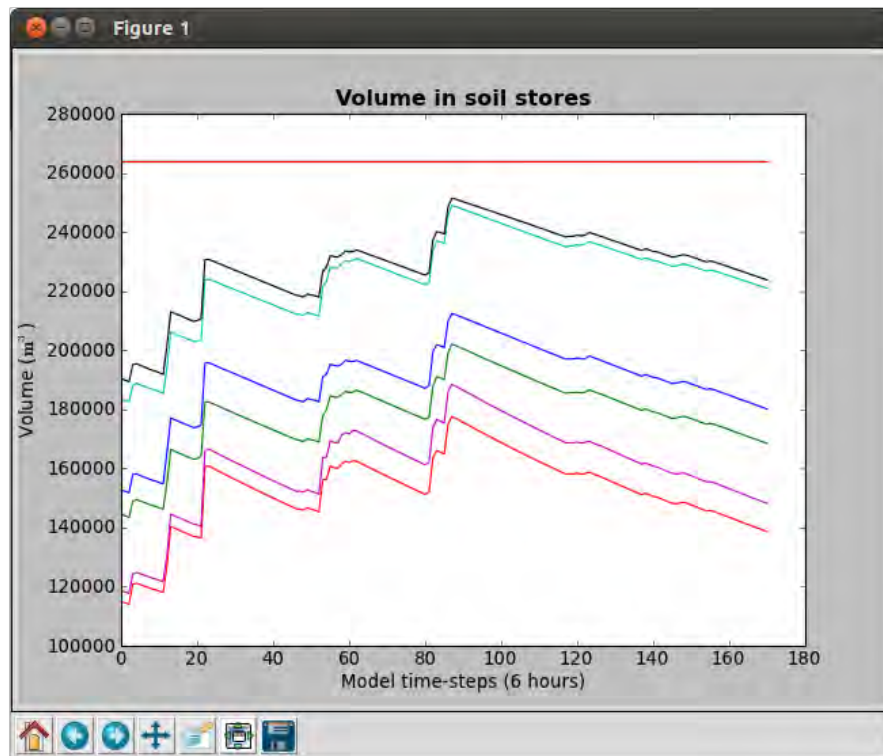
The following figure is produced:



Next we'll plot the time history of the water volume in the several soil stores using:

```
soil_stores = h5file['Soil/V_s'][...]
plt.plot(soil_stores[:, :10])
plt.title('Volume in soil stores', fontweight='bold')
plt.ylabel('Volume ( $\text{m}^3$ )')
plt.xlabel('Model time-steps (6 hours)')
# close simulation file when finished
h5file.close()
```

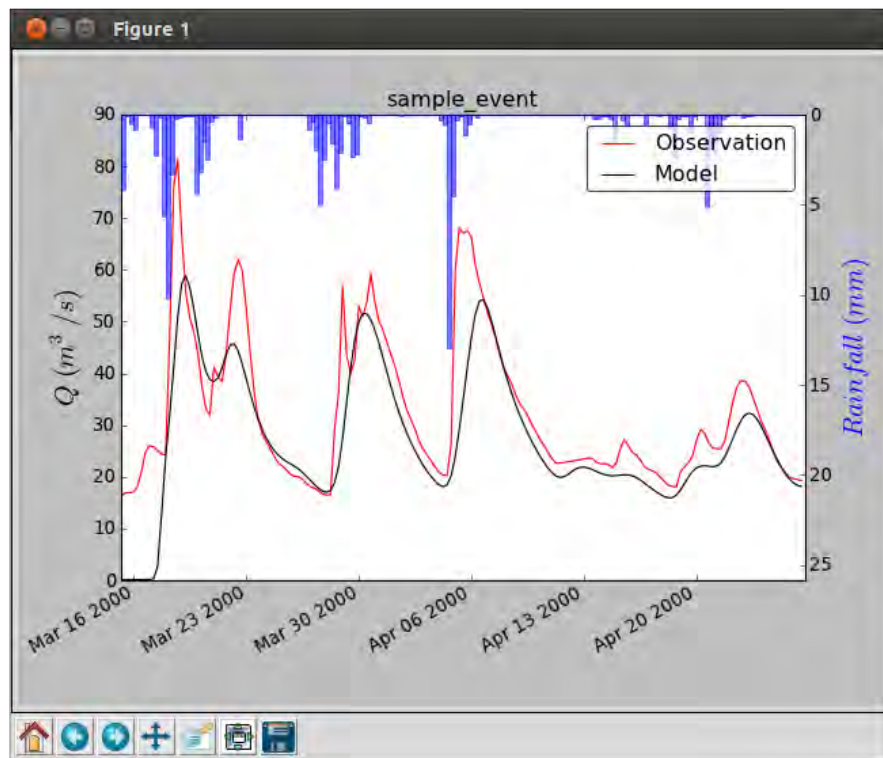
Which produces the following plot:



We've been producing our own plots of the simulation results using the Matplotlib tools. This is very flexible as it allows us to plot what we'd like to see. PyTOPKAPI also has some basic plot helper functions (due to be extended and improved) that can produce plots of the simulation results. As an example running the following:

```
from pytopkapi.results_analysis import plot_Qsim_Qobs_Rain
plot_Qsim_Qobs_Rain.run('plot-flow-precip.ini')
```

Produces the plot window shown here, and also saves a graphic in the *results* sub-directory:



The plot is controlled by the configuration file:

```

plot-flow-precip.ini (~/PyTOPKAPI_workshop/example_simulation) - gedit
Open Save Undo Redo
plot-flow-precip.ini
# This ini files describes the locations of the model input and output
# files and controls some of the options for creating a plot of the
# results (rainfall runoff graphic). Paths may be specified relative
# to the location of the script reading this file or in absolute
# terms. At this stage a '/' must be used as the directory separator
# (even on Windows)

[files]
file_Qsim = results/Example_simulation_results.h5
file_Qobs = forcing_variables/C8H020_6h_Obs_Example.dat
file_rain = forcing_variables/rainfields.h5
image_out = results/Example_simulation_Qsim_Qobs_P_color.png

[groups]
group_name = sample_event

[flags]
Qobs = True
Pobs = True
nash = False

```

The configuration parameters in *plot-result.ini* are:

`file_Qsim = <path to a PyTOPKAPI simulation file>`

```

file_Qobs = <path to a text file containing observed streamflow>
file_rain = <path to PyTOPKAPI rainfall forcing file>
image_out = <file to save the resulting graphic to>

group_name = <name of the simulation group>

Qobs = <flag, plot observed flows if True>
Pobs = <flag, plot observed rainfall if True>
nash = <flag, compute Nash-Sutcliffe efficiency if True>

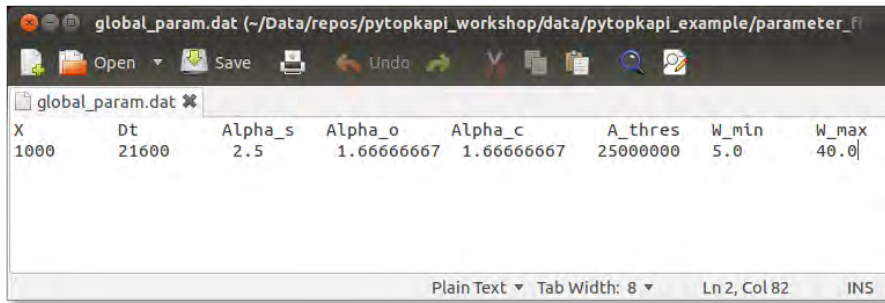
```

3 Description of parameter and forcing data files

In order to apply the PyTOPKAPI model in a new catchment, it's necessary to generate two parameter files as well as data files containing the rainfall and evapotranspiration forcing for the required simulation period (if the catchment is fed by an inter-basin transfer then an additional data file containing these inflows is also required). This section describes the format of the required model input and description files. In section 4 we'll learn to use the tools PyTOPKAPI provides to assist the user in their creation.

Global parameter file

This is a simple ASCII file containing all the parameters that are constant for all cells in the model. The first line gives the names of the variables, with the values reported in the second line.



From left to right in the file the reported parameters are:

X is the lateral dimension of the grid-cell (in m).

Δ_t is the time step of the model (in s)

α_s is a dimensionless pore-size distribution parameter (Brooks and Corey, 1964)

α_o and α_c are the well known power coefficient equal to 5/3 originating from Manning's equation.

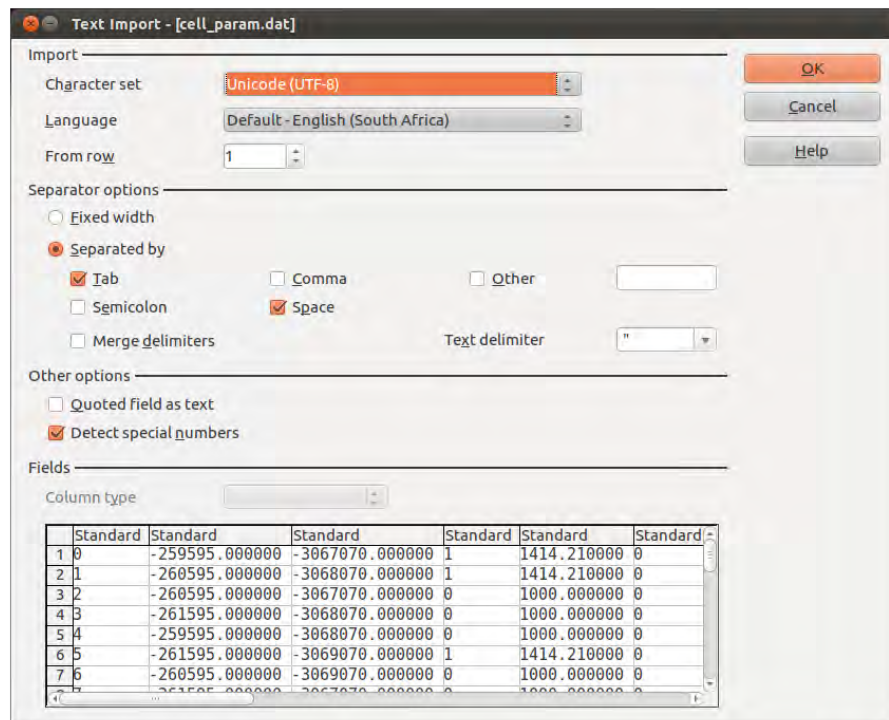
$A_{threshold}$ is the area over which a cell is considered to initiate a river channel (in m^2)

W_{min} is the minimum width of channel (in m)

W_{max} is the maximum width of channel (in m)

Cell parameter file

This is an ASCII file giving the values of spatially variable model parameters. The data are arranged in a tabular format, with each row describing a single cell in the catchment and each of the 21 columns a parameter (described below). You can find a cell parameter file in the *parameter_files* sub-directory. Right click the file *cell_params.dat* and choose the option to *Open with LibreOffice Calc*. Make sure that the check-boxes shown in the graphic below are checked/unchecked.



The spreadsheet should look like this:

The parameters in each column from left to right are:

- **cell_label** - an integer label uniquely identifying each cell.
- **X** - the X location of the cell's centre (in the co-ordinate system selected for modelling)
- **Y** - the Y location of the cell's centre (in the co-ordinate system selected for modelling)
- **channel_cell** - switch to identify channel cells. Equal to 1 if the cell contains a channel, 0 otherwise.
- **X_c** - the length of the channel in cells that have one (in *m*)
- **Dam** - has a value of 1 if the cell is a dam cell, 0 otherwise. (This option is implemented yet, but the space is reserved for possible future developments)
- **tan(β)** - the tangent of the ground slope angle β
- **tan(β_{channel})** - the tangent of the channel slope angle β_{channel}
- **L** - the soil depth (in *m*)
- **K_{SAT}** - the saturated hydraulic conductivity (in *mm/s*)
- **θ_r** - the residual soil moisture content
- **θ_s** - the saturated soil moisture content
- **n_o** - the Mannings roughness coefficient for overland flows
- **n_c** - the Mannings roughness coefficient for the channel flows

- **cell_down** - the label of the downstream cell in the network.
- **pV_s^{t₀}** - the initial saturation of the soil reservoir (in %)
- **V_o^{t₀}** - the initial water content of the overland reservoir (in m³)
- **Q_c^{t₀}** - the initial discharge in the channel (in m³/s)
- **K_c** - the crop coefficient
- **psi_b** - bubbling pressure
- **λ** - pore size distribution

Rainfall forcing file

The rainfall forcing file is an HDF5 binary file containing rainfall at each cell for each time-step in the simulation period. The forcing is stored in a two-dimensional array with each row representing a single time-step and each column a single model cell.

We won't go into any further detail during the workshop - all the rainfall forcing for this tutorial has been prepared ahead of time.

Evapotranspiration forcing file

The evapotranspiration forcing file is an HDF5 binary file with a very similar structure to the rainfall forcing file. The forcing is stored in a two-dimensional array with each row representing a single time-step and each column a single model cell.

The PyTOPKAPI model uses actual evapotranspiration ET_a derived from reference crop evapotranspiration ET_0 using FAO-56 (Allen et al., 1998).

$$ET_a = K_s K_c ET_0 \quad (1)$$

where K_c is a crop factor and K_s is a function of soil water availability. For historical reasons the ET forcing file should currently contain values of $K_c ET_0$ for each cell and time-step. K_s is applied inside the model, based on water volume in the soil store. In future versions of the model, the ET forcing may change.

We won't go into any further detail during the workshop - all the ET forcing for this tutorial has been prepared ahead of time.

4 Applying PyTOPKAPI in a new catchment

Once the two parameter files and forcing data have been prepared, the model can be run as we did in section 2. This section begins by describing the tools available in the PyTOPKAPI package for generating parameter files from gridded parameters over a catchment. The final part is a self-study exercise where each participant must use the parameter files prepared here to run a model simulation with the rainfall and evapotranspiration forcing supplied.

Generating parameter files

The generation of parameter files is probably the most difficult aspect of using PyTOPKAPI, as it requires gridding the physical properties of your catchment as well as determining where to obtain this information in the first place. It is not a process that can be easily automated, and model users will require GIS skills to prepare the required layers.

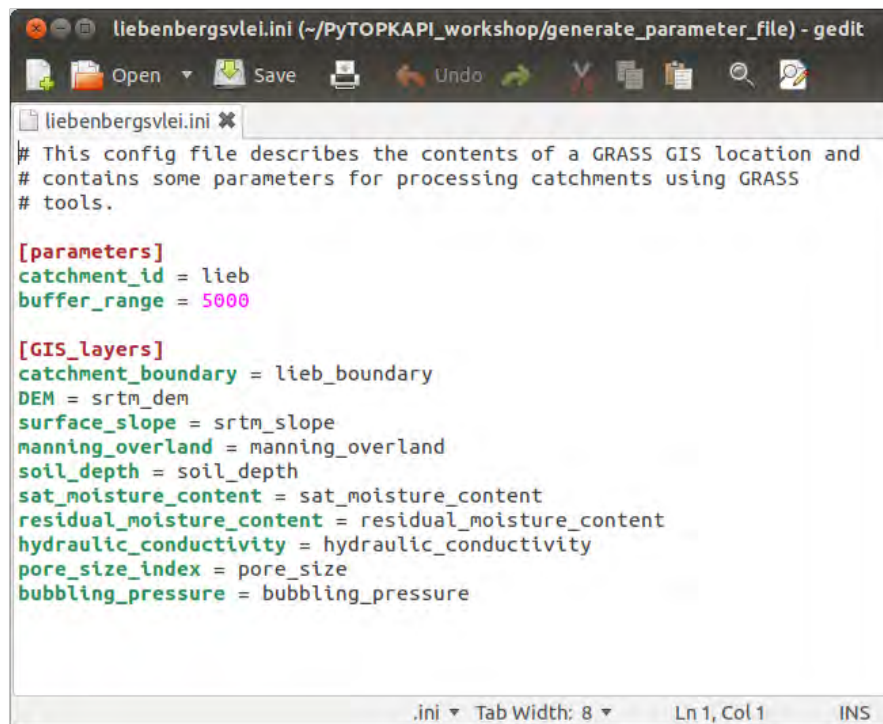
This section describes a tool provided to automate the process of extracting the relevant catchment data from a GRASS GIS location containing pre-gridded catchment data at the spatial resolution and projection required by your project. We are unable to cover the detail of preparing the GRASS location during this tutorial (some of the ideas were covered in the morning lectures).

For users who prefer to use alternative GIS systems the PyTOPKAPI package contains a second tool to generate a parameter file from a set of GeoTIFF files containing the required data for a catchment (the necessary files will be described in the sections that follow). This two stage process separates the work of obtaining and organising the required parameters from the procedure for generating a valid parameter file.

Catchment extraction using GRASS GIS

If you have a GRASS location containing raster maps of the parameters at your required modelling resolution, it is simple to extract a set of GeoTIFF files for the catchment (or sub-catchment) of interest. PyTOPKAPI provides a script called *process_catchment* that uses a catchment boundary vector to extract the necessary model parameters from the raster base maps.

The *process_catchment* script is controlled by a configuration file. The one that we'll use in this exercise is shown below:



```
liebenbergsvlei.ini (-/PyTOPKAPI_workshop/generate_parameter_file) - gedit
# This config file describes the contents of a GRASS GIS location and
# contains some parameters for processing catchments using GRASS
# tools.

[parameters]
catchment_id = lieb
buffer_range = 5000

[GIS_layers]
catchment_boundary = lieb_boundary
DEM = srtm_dem
surface_slope = srtm_slope
manning_overland = manning_overland
soil_depth = soil_depth
sat_moisture_content = sat_moisture_content
residual_moisture_content = residual_moisture_content
hydraulic_conductivity = hydraulic_conductivity
pore_size_index = pore_size
bubbling_pressure = bubbling_pressure

.ini Tab Width: 8 Ln 1, Col 1 INS
```

The configuration file is in the familiar INI format with sections and options as described below:

```
[parameters]
catchment_id = <prefix for naming output files>
buffer_range = <buffer range around boundary vector in map units>
```

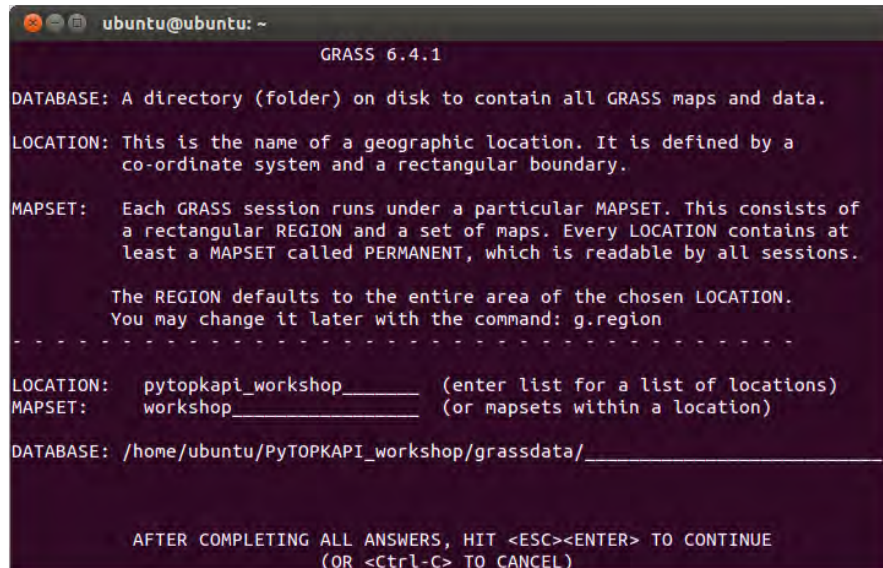
```
[GIS_layers]
catchment_boundary = <name of catchment boundary vector>
DEM = <name of DEM raster>
surface_slope = <name of slope raster>
manning_overland = <name of Manning overland roughness raster>
soil_depth = <name of soil depth raster>
sat_moisture_content = <name of saturated moisture content raster>
residual_moisture_content = <name of residual moisture content raster>
hydraulic_conductivity = <name of hydraulic conductivity raster>
pore_size_index = <name of pore size index raster>
bubbling_pressure = <name of bubbling pressure raster>
```

The easiest way to run the *process_catchment* script is from within a GRASS GIS session, since this will work correctly across all operating systems that can run GRASS and Python. PyTOPKAPI does provide a tool to run the script directly from Python without starting GRASS, but this only works on Linux at this stage.

Since we only plan to run a single command line script, we'll start GRASS in command line mode. To do this, use the system dashboard to open a new terminal window, then type

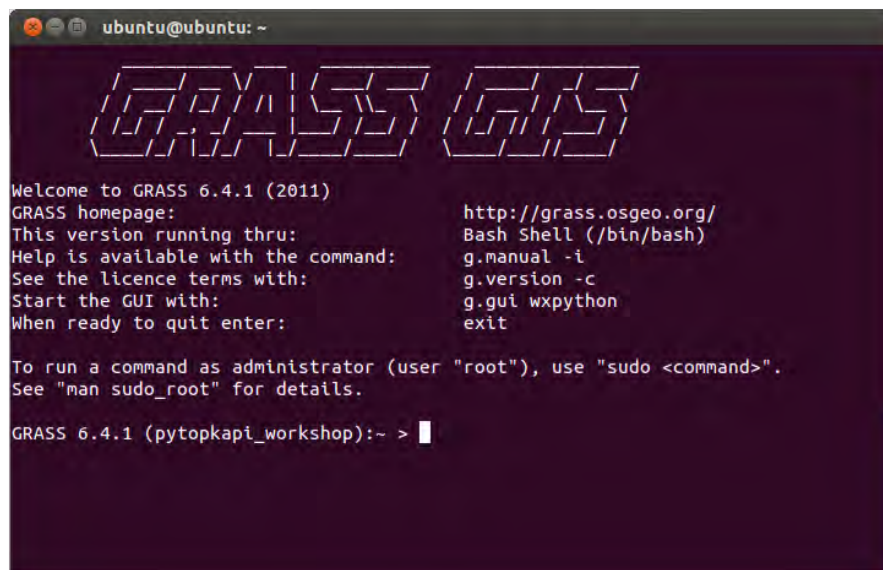
```
$ grass
```

at the command prompt and hit *Enter* (Note - you should not type the \$ sign as this is just a clue that you are in the terminal). The following should appear in your terminal:



```
ubuntu@ubuntu: ~  
GRASS 6.4.1  
DATABASE: A directory (folder) on disk to contain all GRASS maps and data.  
LOCATION: This is the name of a geographic location. It is defined by a  
co-ordinate system and a rectangular boundary.  
MAPSET: Each GRASS session runs under a particular MAPSET. This consists of  
a rectangular REGION and a set of maps. Every LOCATION contains at  
least a MAPSET called PERMANENT, which is readable by all sessions.  
The REGION defaults to the entire area of the chosen LOCATION.  
You may change it later with the command: g.region  
-----  
LOCATION: pytopkapi_workshop_____ (enter list for a list of locations)  
MAPSET: workshop_____ (or mapsets within a location)  
DATABASE: /home/ubuntu/PyTOPKAPI_workshop/grassdata/_____  
  
AFTER COMPLETING ALL ANSWERS, HIT <ESC><ENTER> TO CONTINUE  
(OR <Ctrl-C> TO CANCEL)
```

Accept the default choices by pressing *Esc Enter* (hold down both at the same time). Your GRASS session should then start:



```
ubuntu@ubuntu: ~  
GRASS 6.4.1  
Welcome to GRASS 6.4.1 (2011)  
GRASS homepage: http://grass.osgeo.org/  
This version running thru: Bash Shell (/bin/bash)  
Help is available with the command: g.manual -i  
See the licence terms with: g.version -c  
Start the GUI with: g.gui wxpython  
When ready to quit enter: exit  
  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
GRASS 6.4.1 (pytopkapi_workshop):~ > █
```

The *process_catchment* script can now be run as follows:

```
$ cd ~/PyTOPKAPI_workshop/generate_parameter_file  
$ process-catchment liebenbergsvlei.ini
```

You'll see some output from GRASS and should now have the following GeoTIFF files in your working directory

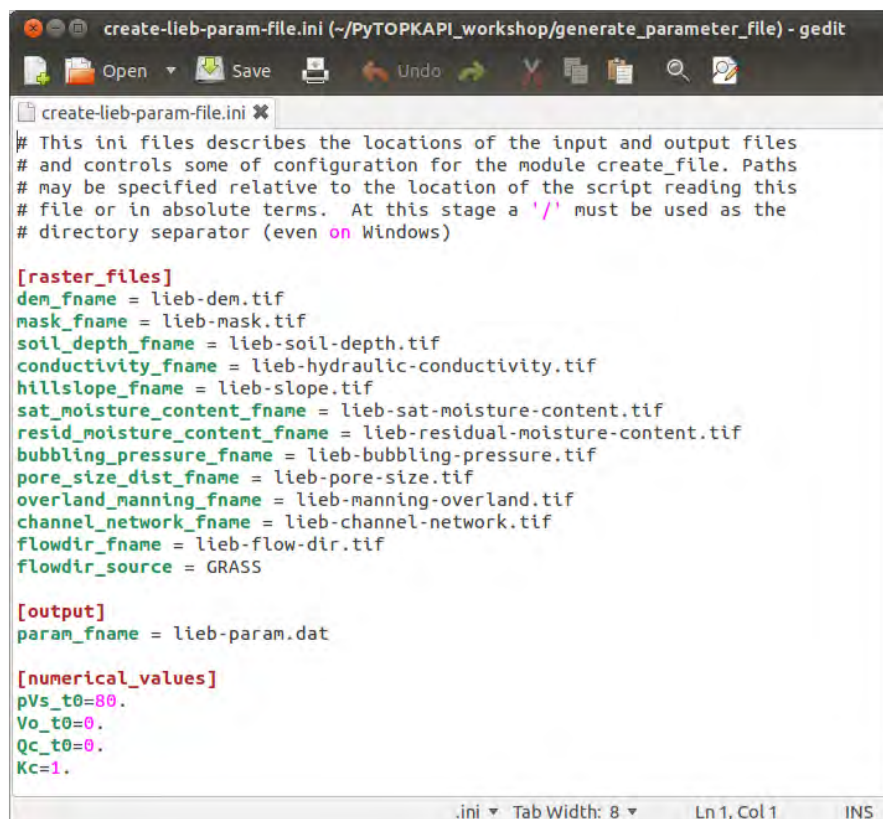

```
lieb-dem.tif
lieb-mask.tif
lieb-flow-dir.tif
lieb-channel-network.tif
lieb-slope.tif
lieb-soil-depth.tif
lieb-sat-moisture-content.tif
lieb-residual-moisture-content.tif
lieb-hydraulic-conductivity.tif
lieb-manning-overland.tif
lieb-bubbling-pressure.tif
lieb-pore-size.tif
```

You are now ready to generate a parameter file from IPython.

Generating a parameter file from GeoTIFFs

The PyTOPKAPI package can generate a properly formatted parameter file, given a set of GeoTIFF files that contain gridded catchment data on a common grid.

The function's behaviour is controlled by a configuration file (*.ini as shown below) with the sections and options described below the figure:



```
create-lieb-param-file.ini
# This ini files describes the locations of the input and output files
# and controls some of configuration for the module create_file. Paths
# may be specified relative to the location of the script reading this
# file or in absolute terms. At this stage a '/' must be used as the
# directory separator (even on Windows)

[raster_files]
dem_fname = lieb-dem.tif
mask_fname = lieb-mask.tif
soil_depth_fname = lieb-soil-depth.tif
conductivity_fname = lieb-hydraulic-conductivity.tif
hillslope_fname = lieb-slope.tif
sat_moisture_content_fname = lieb-sat-moisture-content.tif
resid_moisture_content_fname = lieb-residual-moisture-content.tif
bubbling_pressure_fname = lieb-bubbling-pressure.tif
pore_size_dist_fname = lieb-pore-size.tif
overland_manning_fname = lieb-manning-overland.tif
channel_network_fname = lieb-channel-network.tif
flowdir_fname = lieb-flow-dir.tif
flowdir_source = GRASS

[output]
param_fname = lieb-param.dat

[numerical_values]
pVs_t0=80.
Vo_t0=0.
Qc_t0=0.
Kc=1.
```

```
[raster_files]
dem_fname = <path to DEM file>
mask_fname = <path to catchment mask file>
soil_depth_fname = <path to soil depth file>
conductivity_fname = <path to saturated conductivity file>
hillslope_fname = <path to hill slope file>
sat_moisture_content_fname = <path to saturated moisture content file>
resid_moisture_content_fname = <path to residual moisture content file>
bubbling_pressure_fname = <path to bubbling pressure file>
pore_size_dist_fname = <path to pore size index file>
overland_manning_fname = <path to overland Manning roughness file>
channel_network_fname = <path to channel network file>
flowdir_fname = <path to flow direction file>
flowdir_source = <source of flowdir file. Can be 'GRASS' or 'ARCGIS'>
```

```
[output]
param_fname = <path to output parameter file>
```

```
[numerical_values]
pVs_t0 = <initial percent saturation of soil stores>
Vo_t0 = <initial volume of overland stores>
Qc_t0 = <initial flow rate in channels>
Kc = <crop factor, currently this must be 1.>
```

Note that the initial soil store and overland volumes, as well as the initial channel flow are constant for all cells. Spatially varying values can be assigned by directly manipulating the parameter file, or by using the routines in *pytopkapi.parameter_utils.modify_file*. These tools will also be used to correct inconsistencies in the elevation model used to generate the parameter file.

In order to generate the parameter file from the set of GeoTIFF files, enter the following commands in your IPython session:

```
cd ~/PyTOPKAPI_workshop/generate_parameter_file

from pytopkapi.parameter_utils.create_file import generate_param_file

ini_fname = 'create-lieb-param-file.ini'
generate_param_file(ini_fname)
```

There should now be a parameter file called *lieb-param.dat* in your working directory. Unfortunately, due to the nature of DEMs it is quite common for adjacent channel cells to have equal heights and therefore a channel slope equal to zero. This must be corrected before running the model. To generate another PyTOPKAPI parameter file with corrected channel slopes, we'll run the following in our IPython session to produce a new parameter file named *lieb-param-slope-corrected.dat*:

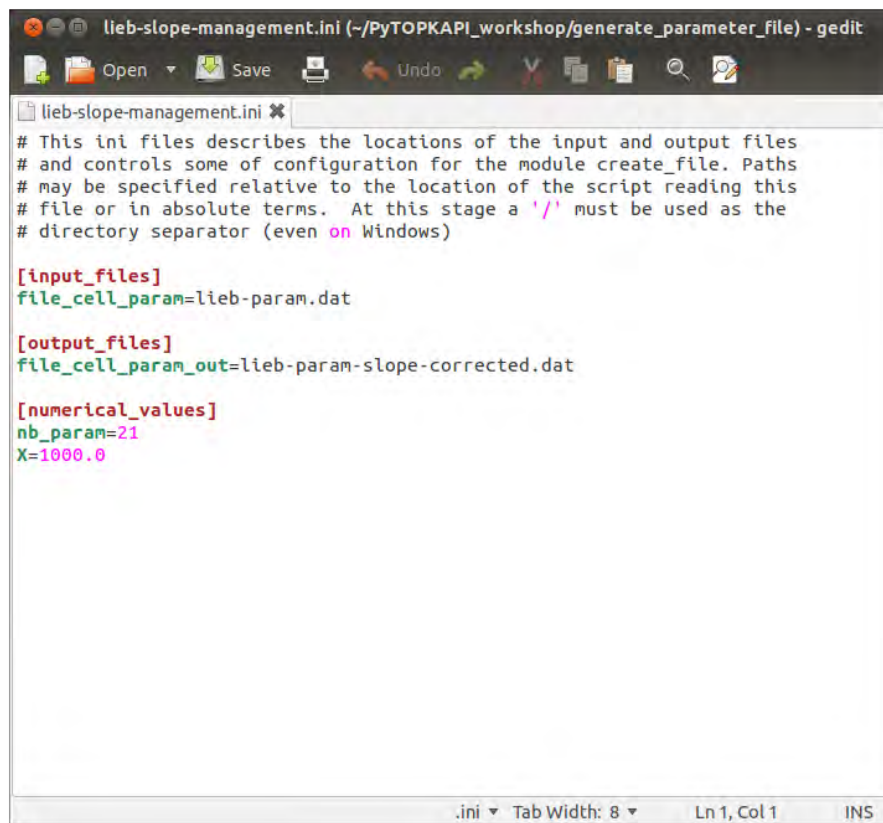
```

from pytopkapi.parameter_utils.modify_file import zero_slope_management

ini_fname = 'lieb-slope-management.ini'
zero_slope_management(ini_fname)

```

As before, the function's behaviour is controlled by a configuration file (*.ini as shown below) with the sections and options described below the figure:



```

[input_files]
file_cell_param = <path to the uncorrected parameter file>

```

```

[output_files]
file_cell_param_out = <path to the channel slope corrected parameter file>

```

```

[numerical_values]
nb_param = <number of model parameters, should always be 21>
X = <length of a model cell in projection units>

```

Putting the techniques into practice

In the final part of this tutorial, participants are asked to use the parameter file generated in the previous section along with the rainfall and ET forcing provided in the *self_study*

directory to run a simulation for the Liebenbergsvlei catchment. Use the files in the *example_simulation* directory as a guide, but note that we will not be including external transfers.

You should try to produce the following figures (we'll discuss the results towards the end of the session):

Fig. 1 Modelled streamflow at the catchment outlet

Fig. 2 Map of soil moisture state for the first 50 time-steps