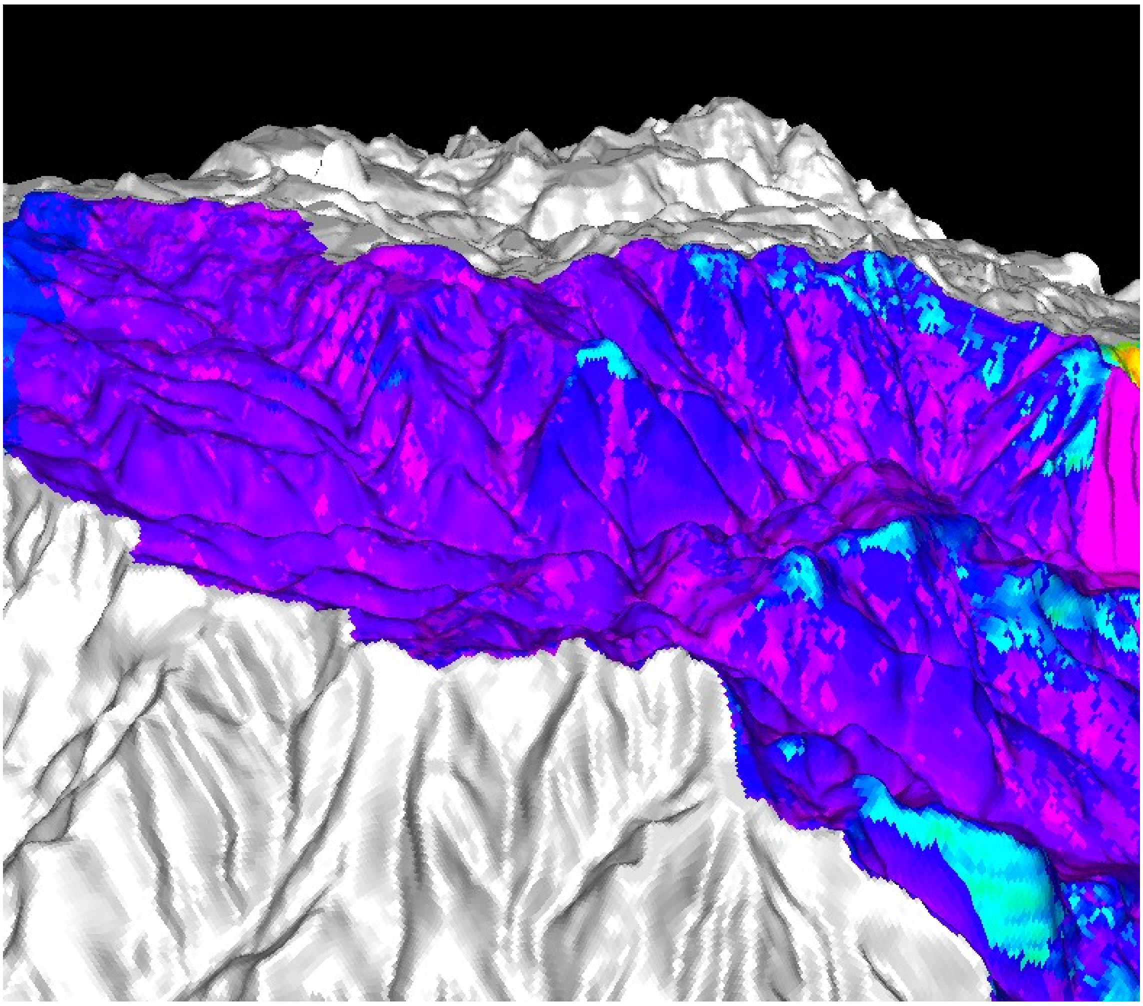
CQ-FLOW: A distributed hydrological model for the prediction of impacts of land-cover change, with special reference to the Rio Chiquito catchment, Northwest Costa Rica



vrije Universiteit amsterdam

FORESTRY RESEARCH PROGRAMME

The Fiesta Project

Version 1.3

J. Schellekens, January 2006

The cqflow model uses code supplied by Mark Mulligan, Arnoud Frumau. Parts of the POTRAD code by Oscar van Dam were used to calculate shade and solar inclination.

TABLE OF CONTENTS

Illustration Index

INTRODUCTION

PURPOSE OF THE MODEL

The main purpose of the `cqflow` model is to quantify the discharge from the Rio Chiquito catchment and evaluate the consequences of several land-use scenarios within the catchment for the Fiesta project.

The model is a research model that may be used for an improved understanding of the processes that are at work in the tropical cloud forest environment. Using the model for other sites and applications is possible but requires the model to be changed and revalidated by an expert user.

STATUS OF THIS DOCUMENT

This document describes the model in detail. However, some parts of the document may be out of date as the model progresses rapidly. In addition, the result of the scenario runs are presented in the final part of this document.

MODEL FRAMEWORK

Development has been done in a framework that allows for rapid development. The main model has been written in `pcraster` and a simple (command-line) user interface has been written in `Octave` (a Freeware Matlab clone).

The model does not include a graphical user interface although it should be possible to run the model from within ArcView/Info using the pcraster ArcView plugin.

PROCESSES INCLUDED IN THE MODEL

Although most of the model is set up in a generic way the model is ***custom made*** for the Rio Chiquito catchment. Specific attention is paid to the effect of inclined precipitation and fog stripping. The following process are included in the model:

1. Based on the location of the site, the time of day and the day of the year the solar inclination is determined. This information is used to re-distribute the point measurements of solar radiation over the DEM.
2. Based on **recorded** horizontal and vertical precipitation inputs the model determines the the angle of precipitation. This is combined with topex and windfield maps (supplied by Mark Mulligan) to scale the precipitation over the entire catchment.
3. Interception is modelled using a simplified version of the Rutter model for rainfall interception that works with hourly time steps. The canopy water budget is solved explicitly.
4. Evaporation for both the dry and wet canopy is determined using the Penman Montieth model. During wet canopy conditions the Rs is set to zero.
5. The soil is represented using a simple bucket model (comparable to the TOPOG_SBM model) that assumes an exponential decay of Ksat with depth. Lateral subsurface flow is modelled using the Darcy equation.
6. A sub-cell parametrization is present that allows a fraction of a cell represent a compacted soil surface with reduced infiltration capacity. This is used to represented the effects of cattle trails
7. Surface runoff is modelled using a kinematic wave routine.

8. The model runs using hourly timesteps.

DATA REQUIREMENTS

Detailed data requirement are described later on in this document. The following list summarizes the data requirements:

- Static data (maps of lookup tables)
  - DEM (resolution <= 150x150 m
  - topex and windfield maps (available for the whole of Costa Rica)
  - Soil Depth
  - Soil physical parameters
  - Land Cover
- Dynamic data (time series)
  - Precipitation (The two separate components of a Juvik type rain gauge)
  - Incoming short-wave radiation
  - Wind speed
  - Wind direction
  - Relative humidity
  - Dry bulb Temperature

LIMITATIONS

- Although the model can be applied to other sites the current code includes many site specific features that require re-coding (in the pcraster script language) if the model is to be applied to other catchments.
- As with any model the accuracy of the results is determined by the quality of the input data and the validity of the simplifications of the actual hydrological processes within the model.
- The model is a research model without a fancy user interface. Operation of the model requires knowledge of the pcraster scripting language.

DESIGN OF THE MODEL

INTRODUCTION

This chapter describes the design of the model and how the model may be adjusted. Installation and the necessary prerequisites for installing the model are described in the User Manual part of this document.

COMPONENTS OF THE MODEL

The `cqflow` model is made up of several components:

1. ***Basemaps preprocessor.*** A script that performs a series of command to prepare the (raw) basemaps for use in the model. Depending on the settings the script reduces the number of cell in the maps to reduce calculation times. This functionality is presently implemented in the `cq_mkbasemaps` Octave script that also calls several pcraster functions.
2. ***Model preprocessor.*** The model preprocessor prepares all the maps needed for the actual model. its most important function is to combines tables with model parameters and soil/vegetation type maps to model parameter maps. This functionality is implemented in the `cqflowmkpars.pcr` pcraster script. command.
3. ***The `cqflow` model.*** This part performs most opf the actual calculations for all the timesteps specified. It is implemented in the `cqflow.pcr` pcraster script.
4. ***Model postprocessor***. Present results of the model run as reports or graphs. Presently some minor postprocessing implemented in Octave/Matlab (simple reports) are available through the model user interface (see below).
5. ***Model user interface.*** Start of the model (and pre/postprocessing utilities) and allow evaluation of model results. A basic (command line) user interface has been made that consists of a number of Octave/Matlab scripts. This interface sets the right command line options for the (pcraster) scrips and allows evaluation of the model results using graphs and reports. The main entry point to the user interface is the `cq_init.m` script that may be run from Octave/Matlab. For the workshop an adapted version of the model has been made that can run from within the pcraster Nutshell environment. Note that this model may yield different results.

Depending on the pre-conditions (has the model been run before or not) different component must be used to perform the model calculation.

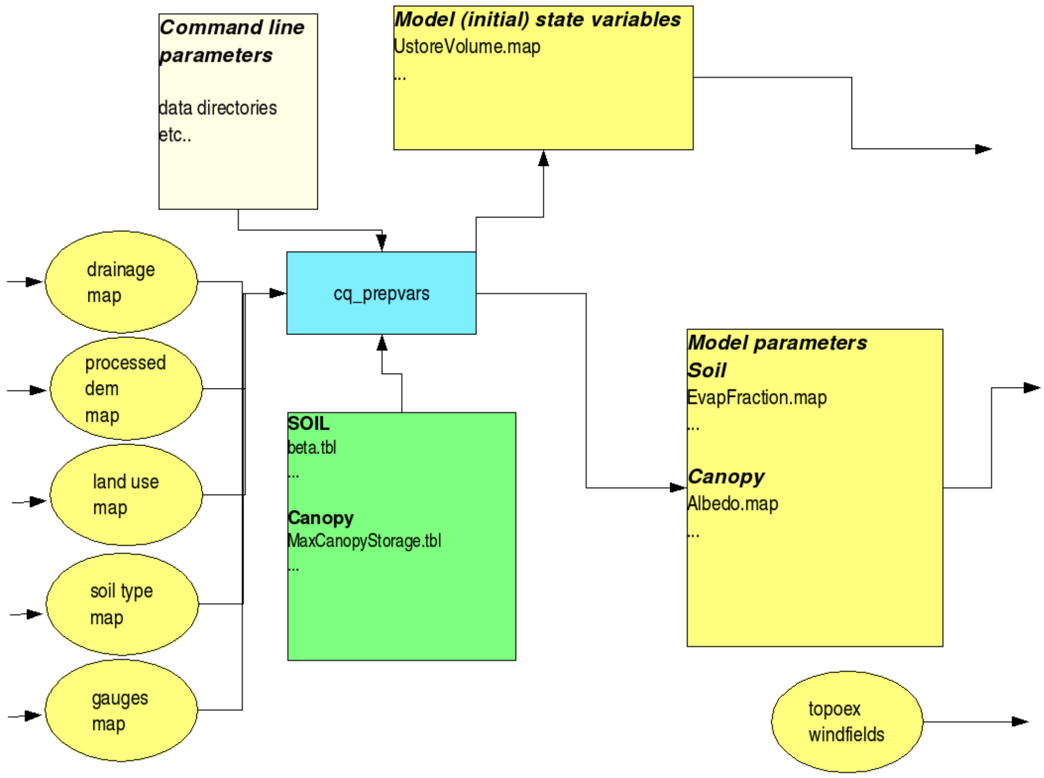


Illustration 1 The cqprepvars script uses look-up tables and the maps from the basemaps pre-processor to construct input maps for the actual model. The arrows coming in from the left show output from the basemaps program while the arrows on the right are maps exported to the actual model.

Basemaps preprocessor

The basemaps preprocessor constructs a number of maps from the basic maps (DEM, soiltype and landuse) that are needed as input to the model. In addition, the script takes location information of gauges to construct sub-catchments and rainfall area polygons (See Figure 2). As data comes from several sources some of the maps are re-sampled to have the same resolution as the dem.

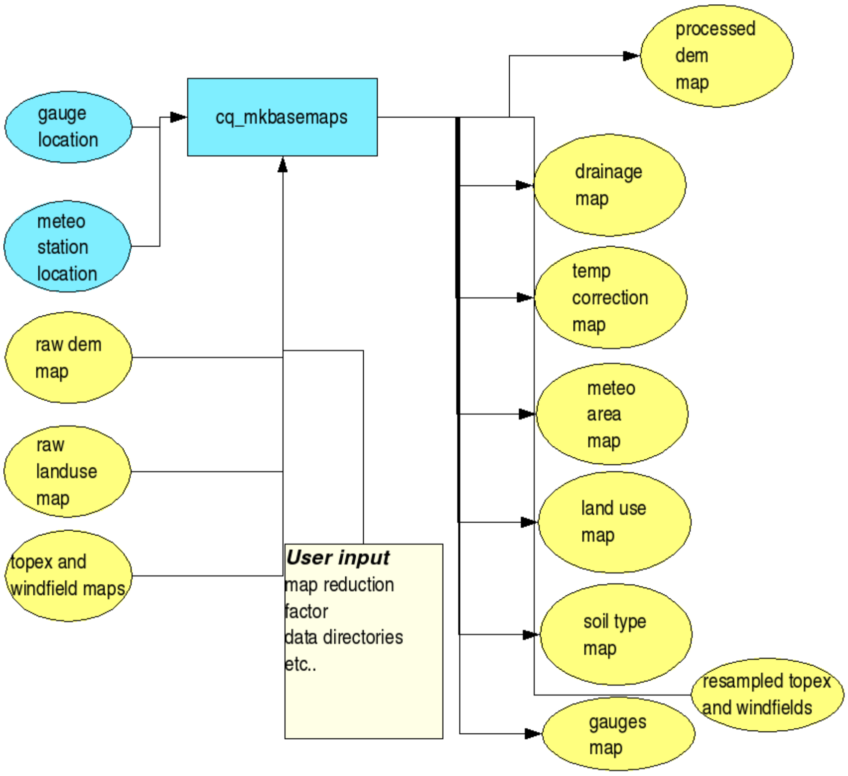


Illustration 2 Data streams of the basemaps pre-processor.

Model preprocessor

The model pre-processor (implemented in the `cqflowmkpars.pcr` script) takes information from ASCII lookup tables and converts those to input maps for the `cqflow` model (see Figure 1). In addition, the script produces default initial state maps.

The cqflow model

The data streams of the cqflow model are shown in Figure 3. The model produces time series (in pcraster .tss format) that are averages over the catchments and maps. These maps are state variables at the end of the run (instantaneous values) and cumulative and average maps of selected variables such as radiation and evaporation. The model can be adjusted to also output maps for each timestep. However, this will use a *lot* of disk-space and will slow down the model considerably.

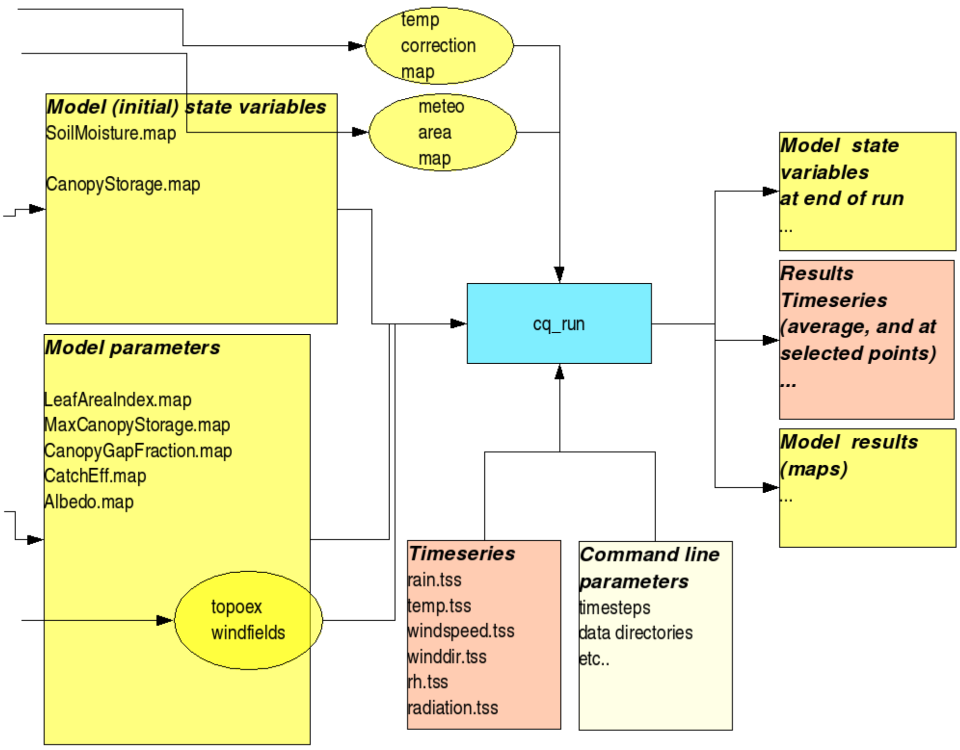


Illustration 3 The data streams for the cqflow model. The model itself is run using the cq_run command in Octave. The model may also be run from an DOS/WINDOWS batch file or a unix shell script.

User interface

The user interface consists of a number of commands implemented in Octave model scripts. Octave is a freeware program that includes most of the functionality of the Matlab program. With some adjustment the interface will also run with Matlab.

MODEL DESCRIPTION AND CALIBRATION

INTRODUCTION

The `cqflow` model was made using parts of several existing models. These were combined to form a new model that is especially suited for the environment and data availability that is found in the Rio Chiquito catchment. The model was written in the PCraster language.

The Solar radiation/inclination functions were taken partly from the POTRAD program by Oscar van Dam and partly from the TAMMOD model by Mark Mulligan.

FLOW CHART OF MODEL CALCULATIONS

The following sequence of of calculations is used within the model:

1. Determine areal average precipitation taking into account the effects of inclined precipitation and the steep terrain.
2. Calculate solar angle and correct point measurements of solar radiation. Distribute over the DEM taking into account shading.
3. Calculate fog interception and add to the canopy storage.
4. Calculate interception losses according to a modified Rutter type model that can work at hourly time steps
5. Perform TOPOG_SBM type soil water and runoff calculations

The sections below elaborate on these steps and describe the `pcraster` code used in the model and the pre-processing phase.

DESCRIPTION OF THE PRE-PROCESSING CODE

The dynamic model uses a number of input maps that are generated from lookup tables. In addition, maps from several sources may have to be re sampled. The pre-processing program takes care of this. The pre-processing is done using a combination of `Octave` and `pcraster` scripts. The first program to call is the `cq_mkbasemaps` program (an octave script) that re samples the basemaps and calculates a number of derived maps.

THE PREPARATION OF THE BASIC MAPS

Preparation of the basic maps requires several pcraster commands and programs. These are all called from the `cq_mkbasemaps` Octave script:

```
function cq_mkbasemaps;
% Creates the basemaps

global CQ_basemaps;

cd(CQ_basemaps);
% Prepares all the static maps needed for the cqflow model
% all other maps are generated from these using the pcraster commands below
```

```
% J. Schellekens 2004
% Many options taken from the scripts prepared by N. Asselman for the
% WL | Delft Hydraulics Economic Benifits of Good watershed Management
% research project.
```

The first step in this script is to resize the DEM (present runs are made using a 150x150 dem) and to create a local drainage network map (LDD) from the DEM.

```
%-------------------------------------------------------------------
% first resize the dem to the desired size (very handy for speeding up
% calculations in the development phase.

%resample -r 4 --unittrue master_dem.map dem.map
reduction = input('Enter the maps reduction factor: ');
pcr_resample(sprintf('-r %d --unittrue master_dem.map dem.map',reduction));


% Now use standard procedure to created the LDD from the DEM
% secondly return the smoothed dem
pcrrun('','ldd.map=lddcreate(dem.map, 1E35, 1E35, 1E35, 1E35)');
pcrrun('','dem.map=lddcreatedem(dem.map, 1E35, 1E35, 1E35, 1E35)');
```

The pcraster streamorder command creates a map of streamorder. This is later used in the model to indicate the main river in the model. Next, several maps are made (these are not used in the model calculations later on) to check the DEM for errors etc.

```
% make streamorder map for later use in kinematic wave
pcrrun('','streamorder.map=streamorder(ldd.map)');
% extract the river for streamorder >= 5
pcrrun('','river.map=streamorder.map ge 5');

% Make a mask for the catchment from the DEM using 1 for the entire area
pcrrun('','catchment.map=boolean(if(dem.map ge -50, 1))');


% Now resample the anneal average rainfall map that Marc M supplied
pcr_resample('--clone dem.map annrainfall.map annualrain.map');


% Now provide a visual check by looking at the upstream area of each point
% Use the natural log to better see what is happening
% this is not used in the mdoel itself
pcrrun('','upstream.map=ln(accuflux(ldd.map, 1))');
```

Based on an ASCII file with the location of the outflow gauge, a map of the subcatchment (upstream area from the gauge) is made. This map is used as a mask in all most other calculations. Presently, the model only works properly with one gauge.

```
% Define subbasins
% - make a .col file with the locations of the gauges
% - these will be converted into a map using the col2map program
col2map('gauges.col gauges.map --clone catchment.map');
pcrrun('','subcatch.map=subcatchment(ldd.map, ordinal(gauges.map))');
pcrrun('','subcatch.map=if(subcatch.map ge 1,subcatch.map)');
```

The model comes with Costa Rica wide maps of topoex and local wind-direction fro the 8 main wind directions. This step extract the data for the Rio Chiquito from these maps. NOTE: Currently (Oct 2005) the Costa Rica maps do not have the right latlong coordinates. Therefore the model uses high resolution 25x25 meter maps prepared for the Rio Chiquito and re samples those to the model resolution.

```
% Extract the Rio C atchment from the Costa rica wind maps
pcr_resample('--clone subcatch.map CQMaps/topexn.map topexn.map');
pcr_resample('--clone subcatch.map CQMaps/topexne.map topexne.map');
pcr_resample('--clone subcatch.map CQMaps/topexe.map topexe.map');
pcr_resample('--clone subcatch.map CQMaps/topexse.map topexse.map');
pcr_resample('--clone subcatch.map CQMaps/topexs.map topexs.map');
pcr_resample('--clone subcatch.map CQMaps/topexsw.map topexsw.map');
```

```
pcr_resample('--clone subcatch.map CQMaps/topexw.map topexw.map');
pcr_resample('--clone subcatch.map CQMaps/topexnw.map topexnw.map');


pcr_resample('--clone subcatch.map CQMaps/windtopon_ldd.map windtopon_ldd.map');
pcr_resample('--clone subcatch.map CQMaps/windtopone_ldd.map windtopone_ldd.map');
pcr_resample('--clone subcatch.map CQMaps/windtopoe_ldd.map windtopoe_ldd.map');
pcr_resample('--clone subcatch.map CQMaps/windtopose_ldd.map windtopose_ldd.map');
pcr_resample('--clone subcatch.map CQMaps/windtopos_ldd.map windtopos_ldd.map');
pcr_resample('--clone subcatch.map CQMaps/windtoposw_ldd.map windtoposw_ldd.map');
pcr_resample('--clone subcatch.map CQMaps/windtopow_ldd.map windtopow_ldd.map');
pcr_resample('--clone subcatch.map CQMaps/windtoponw_ldd.map windtoponw_ldd.map');
```

Other maps (e.g. Several land use scenarios) are re-sampled to the resolution of the DEM.

```
% now resample the vegetation/landuse maps to the size etc of the dem
pcr_resample('--clone subcatch.map lu2001.map landuse.map');
pcr_resample('--clone subcatch.map lu1975.map landuse1975.map');
pcr_resample('--clone subcatch.map lu1987.map landuse1987.map');
pcr_resample('--clone subcatch.map Landuse0.001 landuse1.map');
pcr_resample('--clone subcatch.map Landuse0.002 landuse2.map');
pcr_resample('--clone subcatch.map Landuse0.003 landuse3.map');
pcr_resample('--clone subcatch.map Landuse0.004 landuse4.map');

pcrrun('','landuse.map=if(boolean(subcatch.map) then landuse.map else 0)');
pcrrun('','landuse1975.map=if(boolean(subcatch.map) then landuse1975.map else 0)');
pcrrun('','landuse1987.map=if(boolean(subcatch.map) then landuse1987.map else 0)');
pcrrun('','landuse1.map=if(boolean(subcatch.map) then landuse1.map else 0)');
pcrrun('','landuse2.map=if(boolean(subcatch.map) then landuse2.map else 0)');
pcrrun('','landuse3.map=if(boolean(subcatch.map) then landuse3.map else 0)');
pcrrun('','landuse4.map=if(boolean(subcatch.map) then landuse4.map else 0)');


% Reshuffle the classes remove 1 and 2 others should now be 1,2,3 missind
% data set to grass
pcrrun('','landuse.map=nominal(if (scalar(landuse.map) - 2 >= 1 then scalar(landuse.map)
         - 2 else 1))');
pcrrun('','landuse1975.map=nominal(if (scalar(landuse1975.map) - 2 >= 1 then
         scalar(landuse1975.map) - 2 else 1))');
pcrrun('','landuse1987.map=nominal(if (scalar(landuse1987.map) - 2 >= 1 then
         scalar(landuse1987.map) - 2 else 1))');



% make sure no missing values are present, fill them with class 1
pcrrun('','landuse.map=cover(landuse.map,nominal((scalar(subcatch.map)*0.0)+1.0))');
pcrrun('','landuse1.map=cover(landuse1.map,nominal((scalar(subcatch.map)*0.0)+1.0))');
pcrrun('','landuse2.map=cover(landuse2.map,nominal((scalar(subcatch.map)*0.0)+1.0))');
pcrrun('','landuse3.map=cover(landuse3.map,nominal((scalar(subcatch.map)*0.0)+1.0))');
pcrrun('','landuse4.map=cover(landuse4.map,nominal((scalar(subcatch.map)*0.0)+1.0))');
pcrrun('','landuse1975.map=cover(landuse1975.map,nominal((scalar(subcatch.map)*0.0)+1.0))
         ');
pcrrun('','landuse1987.map=cover(landuse1987.map,nominal((scalar(subcatch.map)*0.0)+1.0))
         ');



% create soiltype from landuse at present
pcrrun('','soiltype.map=landuse.map');
```

From an ASCII file that contains a list of weather station locations a map is made of the area for each gauge. In addition, these locations are used to create a temperature correction map.

```
% Now make the metgauge areas map
col2map('--clone catchment.map -O  meteostat.col meteostat.map');
%## Determine disctance to station for each c ell
```

```
pcrrun('','meteoarea.map=if(dem.map ge -50,spreadzone(cover(meteostat.map,0),0,1))');


% Temperature correction for altitude using 0.0006 degC/m laps rate
% NOTE: This procedure does _NOT_ work for stations outside the DEM!!

pcrrun( '','meteoalt.map=if(boolean(meteostat.map) then dem.map)');
pcrrun( '','meteoalt.map=areaaverage(meteoalt.map, meteoarea.map)');
pcrrun( '','altcortemp.map=(0.0006*(meteoalt.map+dem.map))');


% Now make a correction map for the Precipitation using the annual map that Marc supplied
         (not used in the actual model)
% determine average annual P at the meteo stations
pcrrun( '','precipann.map=if(boolean(meteostat.map) then annualrain.map)');
pcrrun( '','precipann.map=areaaverage(precipann.map, meteoarea.map)');
pcrrun( '','precipcorr.map=(precipann.map/annualrain.map)');


% Now copy the maps we need to the staticmaps directory
copyfile('meteostat.map','../staticmaps/');
copyfile('meteoarea.map','../staticmaps/');
copyfile('dem.map','../staticmaps/');
copyfile('ldd.map','../staticmaps/');
copyfile('landuse.map','../staticmaps/');
copyfile('soiltype.map','../staticmaps/');
copyfile('subcatch.map','../staticmaps/');
copyfile('altcortemp.map','../staticmaps/');
copyfile('gauges.map','../staticmaps/');
copyfile('river.map','../staticmaps/');

copyfile('topexn.map','../staticmaps/');
copyfile('topexe.map','../staticmaps/');
copyfile('topexs.map','../staticmaps/');
copyfile('topexw.map','../staticmaps/');
copyfile('topexne.map','../staticmaps/');
copyfile('topexnw.map','../staticmaps/');
copyfile('topexse.map','../staticmaps/');
copyfile('topexsw.map','../staticmaps/');

copyfile('windtopon_ldd.map','../staticmaps/');
copyfile('windtopoe_ldd.map','../staticmaps/');
copyfile('windtopos_ldd.map','../staticmaps/');
copyfile('windtopow_ldd.map','../staticmaps/');
copyfile('windtopone_ldd.map','../staticmaps/');
copyfile('windtoponw_ldd.map','../staticmaps/');
copyfile('windtopose_ldd.map','../staticmaps/');
copyfile('windtoposw_ldd.map','../staticmaps/');

cd ..
```

Preparing parameter and initial state maps

Default initial state values and parameter maps are made using the `cq_prepvars` program. This program does not do any calculations itself but it calls the `cqflowmkpars.pcr pcraster` script. This script is shown below:

```
# Preprocessor for the cqflow model

# J. Schellekens
# Arguments:

# pcrcalc -f cvqflowmkpars.pcr  basemaps  intabels generated_maps
# basemaps = directory with (static) basemaps (DEM etc)
# intabels = directory with look-up tables with parameters
# generated_maps = dir to put the generated maps in. The model gets
#                          the data from this directory
```

Bind the base maps used to derive the parameters maps.

```
binding

Altitude=$1\dem.map;            # DEM
TopoLdd=$1\ldd.map;             # Local drain Direction map
TopoId=$1\subcatch.map;         # area map
OutputId=$1\gauges.map;         # location of river gaugingstation
SoilType=$1\soiltype.map;       #
LandUse=$1\landuse.map;         # landuse map
```

Topog SBM and other soil parameters

```
# TOPOG_SBM type soil stuff
KsatHorTbl =$2\KsatHor.tbl;                     # Horizontal Ksat in mm/hr
FirstZoneKsatVerTbl = $2\FirstZoneKsatVer.tbl;  # Vertical Ksat in mm/hr
BetaTbl = $2\Beta.tbl;                          # =0.6;
# Mannings N. Set this higher to get more dampening of the discharge peak
NTbl=$2\N.tbl;# =0.2;
FirstZoneCapacityTbl=$2\FirstZoneCapacity.tbl; # [mm]
InfiltCapSoilTbl=$2\InfiltCapSoil.tbl; # [mm/hr]
InfiltCapPathTbl=$2\InfiltCapPath.tbl; #  [mm/hr]
PathFracTbl=$2\PathFrac.tbl; # Fraction of compacted areas/paths in the cell

# Parameters and initial for the TOPOG SBM type soil model

thetaSTbl=$2\thetaS.tbl;                    #  0.6  saturated water content
thetaRTbl=$2\thetaR.tbl;                    #  0.1  residual water content
MTbl=$2\M.tbl;                              #  220 Scaling parameter


# For vegetation the following maps should be made
# - LAI
# - Canopy storage
# - Canopy Gap fraction
LeafAreaIndexTbl=$2\LeafAreaIndex.tbl;
MaxCanopyStorageTbl=$2\MaxCanopyStorage.tbl;
CanopyGapFractionTbl=$2\CanopyGapFraction.tbl;

# these are not used anymore...
foneTbl=$2\fone.tbl; # Table with regression constats to estimate Rs
ftwoTbl=$2\ftwo.tbl; # Table with regression constats to estimate Rs
RsTbl=$2\Rs.tbl;
RaTbl=$2\Ra.tbl;
#WindSpeedHeigthTbl=$2\WindSpeedHeigth.tbl;
AlbedoTbl=$2\Albedo.tbl;
RootingDepthTbl = $2\RootingDepth.tbl;      # =1000;
VegetationHeigthTbl=$2\VegetationHeigth.tbl;
# The following are set here for the fog interception model
CatchEffTbl=$2\CatchEff.tbl; # Maps the land-suse type to a fog catching efficiency

areamap
 TopoId;


initial

ZeroMap=0*scalar(TopoId);                                #map with only zero's

# Link Tables to soil parameters.....

#Soil type linked parameters
KsatHor=lookupscalar(KsatHorTbl,SoilType);             #
FirstZoneKsatVer=lookupscalar(FirstZoneKsatVerTbl,SoilType);    #
Beta=lookupscalar(BetaTbl,SoilType);      #
```

```
M=lookupscalar(MTbl,SoilType);     #Manning N
N=lookupscalar(NTbl,SoilType);               #Scaling parameter
thetaS=lookupscalar(thetaSTbl,SoilType);               #Scaling parameter
thetaR=lookupscalar(thetaRTbl,SoilType);               #Scaling parameter

# Link tables to landuse map to geCanopyStoraget vegetation parameters
LeafAreaIndex=lookupscalar(LeafAreaIndexTbl,LandUse);
MaxCanopyStorage=lookupscalar(MaxCanopyStorageTbl,LandUse);
CanopyGapFraction=lookupscalar(CanopyGapFractionTbl,LandUse);
Albedo=lookupscalar(AlbedoTbl,LandUse);
Rs=lookupscalar(RsTbl,LandUse);
Ra=lookupscalar(RaTbl,LandUse);
RootingDepth=lookupscalar(RootingDepthTbl,LandUse);
InfiltCapSoil=lookupscalar(InfiltCapSoilTbl,LandUse);
InfiltCapPath=lookupscalar(InfiltCapPathTbl,LandUse);
PathFrac=lookupscalar(PathFracTbl,LandUse);
fone=lookupscalar(foneTbl,LandUse);
ftwo=lookupscalar(ftwoTbl,LandUse);

VegetationHeigth=lookupscalar(VegetationHeigthTbl,LandUse);

# Fog interception efficiency
CatchEff=lookupscalar(CatchEffTbl,LandUse);

# Soil moisture state for new model
FirstZoneCapacity = lookupscalar(FirstZoneCapacityTbl,LandUse);


FirstZoneDepth=ZeroMap + 0.8 * FirstZoneCapacity;
UStoreDepth = 0.8 * (FirstZoneCapacity - FirstZoneDepth);
SurfaceRunoff=ZeroMap + 0.00000001;

# Prepare default state variables
CanopyStorage=ZeroMap;
# watercontent map generated here, this is temporary
WaterContent=if (Altitude < 1050 then ZeroMap else ZeroMap + 0.3);

# save state variables
report $3\CanopyStorage.map=CanopyStorage;
report $3\WaterContent.map=WaterContent;
report $3\FirstZoneDepth.map=FirstZoneDepth;
report $3\UStoreDepth.map=UStoreDepth;
report $3\SurfaceRunoff.map=SurfaceRunoff;






# save soil parameter maps
report $3\KsatHor.map=KsatHor;
report $3\FirstZoneKsatVer.map=FirstZoneKsatVer;
report $3\Beta.map=Beta;
report $3\N.map=N;
report $3\thetaS.map=thetaS;
report $3\thetaR.map = thetaR;
report $3\M.map=M;
report $3\FirstZoneCapacity.map=FirstZoneCapacity;
report $3\InfiltCapSoil.map=InfiltCapSoil;
report $3\InfiltCapPath.map=InfiltCapPath;
report $3\PathFrac.map=PathFrac;






# Save vegetation parameter maps
report $3\LeafAreaIndex.map=LeafAreaIndex;
report $3\MaxCanopyStorage.map=MaxCanopyStorage;
report $3\CanopyGapFraction.map=CanopyGapFraction;
```

```
report $3\Albedo.map=Albedo;
report $3\Rs.map=Rs;
report $3\Ra.map=Ra;
report $3\CatchEff.map=CatchEff;
report $3\RootingDepth.map=RootingDepth;
report $3\fone.map=fone;
report $3\ftwo.map=ftwo;
report $3\VegetationHeigth.map=VegetationHeigth;
```

DESCRIPTION OF THE CQFLOW CODE AND RESULTS

THE BINDING SECTION

In this first section of the program internal variables are linked to input maps and time series on the file system. The $1 to $6 items represent the command-line arguments that are passed to the program. These determine the directories in which input/output maps and time series are located. Within `pcraster` the sharp sign (#) is used to start a comment. In the first part the static maps are linked to model variables:

```
binding

###########################################################################
# Input base maps #########################################################
###########################################################################
Altitude=$6\dem.map;                # DEM
TopoLdd=$6\ldd.map;                # Local
TopoId=$6\subcatch.map;    # area map
River=$6\river.map;
subcatch=$6\subcatch.map;
LandUse=$6\landuse.map;
RainAreas=$6\meteoarea.map;  # Raingauge/tempgauge zones.
                                    # Number of areas should match the columns in
                                    # the rain.txt input file
MeteoStations=$6\meteostat.map;     # Actual locations of meteo stations

OutputLoc=$6\gauges.map;         # location of output gauge(s)
OutputId=$6\subcatch.map;        # location of subcatchment
Alt2Temp=$6\altcortemp.map;         # temperature compensation for elevation differences
```

Next the topex and wind-direction maps are linked to variables.

```
# TOPEX and Wind direction maps for all eight major directions
# These are extracted from the map for the whole of Costa Rica prepared by
# Marc Mulligan in the cq_mkbasemaps script.
topexn = $6\topexn.map;
topexne = $6\topexne.map;
topexe = $6\topexe.map;
topexse = $6\topexse.map;
topexs = $6\topexs.map;
topexsw = $6\topexsw.map;
topexw = $6\topexw.map;
topexnw = $6\topexnw.map;

windn = $6\windtopon_ldd.map;
windne = $6\windtopone_ldd.map;
winde = $6\windtopoe_ldd.map;
windse = $6\windtopose_ldd.map;
winds = $6\windtopos_ldd.map;
windsw = $6\windtoposw_ldd.map;
windw = $6\windtopow_ldd.map;
windnw = $6\windtoponw_ldd.map;
```

In the next part the input time series are linked to model variables:

```
###########################################################################
# Input time series #######################################################
###########################################################################
RainTss=$3\rain.tss;                # timeseries for rainfall
TempTss=$3\temp.tss;                # timeseries for temperature
RHTss=$3\rh.tss;                 # timeseries for relative Humidity
ShortWaveTss=$3\rad.tss;         # Measured short wave radiation
WindSpeedTss=$3\windspeed.tss;   # input timeseries with windspeed (NB
                                    # assumed to be located at temp/rain stations)
WindDirTss=$3\winddir.tss;          # input timeseries with wind direction (NB
                                    # assumed to be located at temp/rain stations)
HPTss=$3\hp.tss;                 # input of Juvic gauge HP component
DayTime=$3\time.tss;             # file with date/time: step, day(1:365), hr (0:24)
```

Further on maps with model parameters are defined:

```
##########################################################################
# Input of parameter Maps ################################################
##########################################################################
# Vegetation parameter maps
LeafAreaIndex=$1\LeafAreaIndex.map;
Cmax=$1\MaxCanopyStorage.map;
CanopyGapFraction=$1\CanopyGapFraction.map;
Ra=$1\Ra.map;
Rs=$1\Rs.map;
fone=$1\fone.map;
ftwo=$1\ftwo.map;
Albedo=$1\Albedo.map;
RootingDepth=$1\RootingDepth.map;
VegetationHeigth=$1\VegetationHeigth.map;
# Fog interception parameters
CatchEff=$1\CatchEff.map;

# SBM soil parameters
FirstZoneCapacity=$1\FirstZoneCapacity.map; # thickness of the soil
KsatHor = $1\KsatHor.map;
InfiltCapSoil = $1\InfiltCapSoil.map;       # infiltration capacity if the soil [mm/hr]
InfiltCapPath = $1\InfiltCapPath.map;       # infiltration capacity of the compacted
                                            # areas (paths) in [mm/hr]
PathFrac = $1\PathFrac.map;        # Fraction area with compacted soil (Paths etc.)
FirstZoneKsatVer = $1\FirstZoneKsatVer.map;
Beta= $1\Beta.map;
N=$1\N.map;
thetaS=$1\thetaS.map;
thetaR=$1\thetaR.map;
M=$1\M.map;
```

In the next part the initial state variable are read from disk and assigned to model variables. Usually these are saved at the end of a previous run.

```
##########################################################################
# Input of initial state maps ############################################
##########################################################################
# read soil initial state maps
FirstZoneDepth=$1\FirstZoneDepth.map;
UStoreDepth=$1\UStoreDepth.map;
SurfaceRunoff=$1\SurfaceRunoff.map;
SecondZoneDepth=$1\SecondZoneDepth.map;
CanopyStorage=$1\CanopyStorage.map;
```

In the final part of the binding section some values are set that do not change during a model run such as the latitude and longitude of the location.

```
##########################################################################
# Set static initial values here #########################################
##########################################################################
timestepsecs=3600;
MaxLeakage = 3.0; # Maximum leakage in mm/day
CriticalAngle = 85; # Seperation angle between HP and P
SoilAlbedo = 0.1; # Not used at the moment
RiverWidth = 10.0;
pi = 3.1416;
e = 2.7183;
k2 = 0.1681;      # (von Karman constant)^2
```

```
Latitude  =  10.359;
Longitude =  84.8014;

SumRad = 0;
Sc       = 1367.0;          # Solar constant (Gates, 1980) [W/m2]
Trans    = 0.6;             # Transmissivity tau (Gates, 1980)
PRadY = 0; PRadM = 0; PRadMD = 0; PRadD = 0;
```

Timer and areamap

The areamap and time section determine the mask when creating new maps and the number of time steps of the model respectively. The number of timesteps is given to the model as a command-line parameter (number 5).

```
areamap
TopoId;

timer
1 $5 1;
```

The initial section

In the initial section static calculations and initializations are performed. All these calculations are performed only once, at the start of a run.

```
############################################################################
# Initial settings and calculations ######################################
############################################################################
initial

# Initializing of variables
TopoLdd=lddmask(TopoLdd,TopoId);

Sold = 0; SoldCor =0;
# General settings

timestepdays=timestepsecs/(60*60*24);
catchmentcells=maptotal(scalar(TopoId));
AtmPcor = ((288-0.0065*Altitude)/288)**5.256;            # atm pressure corr [-]

# temporary needed to get outputs for above 1400m seperately
#OutZones= nominal(if(Altitude > 1400 then scalar(LandUse) + 3 else scalar(LandUse)));
OutZones = LandUse;

# fraction of water that may re-infiltrate if a part of a cell has a compacted area
ReInfiltrateFrac = 0.5;

# Determine real slope.
reallength=celllength()/0.00022483 * 25; # convert lat/long to meters to get slope right.
Slope=max(0.00001,slope(Altitude)*celllength()/reallength);
Terrain_angle=scalar(atan(Slope));

# Convert Q
QMMConv = (reallength * reallength * 0.001 * catchmentcells);

#report xx=QMMConv;
# report $2\size.map=reallength * reallength  * catchmentcells;

UpstreamElements=accuflux(TopoLdd,1);  #number of upstream elements
ZeroMap=0*scalar(subcatch);                    #map with only zero's
sumprecip=0;                           #accumulated rainfall for water balance
sumevap=0;                             #accumulated evaporation for water balance
sumrunoff=0;                           #accumulated runoff for water balance
sumint=0;                              #accumulated interception for water balance
sumleakage=0;
sumoutflow=0;

CumCorrectedExposureWindSpdRatio =ZeroMap;
```

```
CumExposureWindSpdRatio=ZeroMap;
CumWindSpeed=ZeroMap;
CumWindSpeedStation=ZeroMap;
CumPrec=ZeroMap;
CumHPPrecAngle=ZeroMap;
CumPrecAngle=ZeroMap;
CumPotPrec=ZeroMap;
CumInfiltExcess=ZeroMap;
CumExfiltWater=ZeroMap;
CumSurfaceWater=ZeroMap;
CumReinfilt=ZeroMap;
CumEvap=ZeroMap;
CumPotenEvap=ZeroMap;
CumInt=ZeroMap;
CumRad=ZeroMap;
CumWaterCatch=ZeroMap;
CumLeakage=ZeroMap;
CumPrecPol=ZeroMap;
CumPrecWindCor=ZeroMap;
FirstZoneFlux=ZeroMap;
FreeWaterDepth=ZeroMap;
SumCellWatBal=ZeroMap;
PathInfiltExceeded=ZeroMap;
SoilInfiltExceeded=ZeroMap;

Aspect=scalar(aspect(Altitude));# aspect [deg]
Aspect  = if(Aspect le 0 then scalar(0.001) else Aspect);
# On Flat areas the Aspect function failes, fill in with average...
Aspect = if (defined(Aspect) then Aspect else areaaverage(Aspect,TopoId));

# TOPOG_SBM type soil stuff
#f = (thetaS -thetaR)/M;
f = 1/M;


UnsatStore = FirstZoneCapacity - FirstZoneDepth;

DCL=max(downstreamdist(TopoLdd),reallength);
H=scalar(0.000000001);    # initial waterheight (m)

# set width for kinematic wave to cell width for all cells
Bw=reallength;
# However, in the main river we have real flow so set the width to the
# width of the river, 10 meter in this case
Bw=if(River then RiverWidth else Bw);
# term for Alpha
AlpTerm=(N/(sqrt(Slope)))**Beta;
# power for Alpha
AlpPow=(2/3)*Beta;
# initial approximation for Alpha
P=Bw+2*H;
Alpha=AlpTerm*(P**AlpPow);

SurfaceRunoffMM=SurfaceRunoff *  timestepsecs/(reallength * reallength * 0.001);

initstorage=areaaverage(FirstZoneDepth,TopoId)+areaaverage(UStoreDepth,TopoId)+areaaverag
         e(CanopyStorage,TopoId)+areaaverage(SurfaceRunoffMM,TopoId);

CellStorage = FirstZoneDepth + UStoreDepth;

zi = FirstZoneCapacity - FirstZoneDepth;#/(thetaS -thetaR);
# Determine actual water depth

########### END OF THE INITIAL SECTION ##################################
```

THE DYNAMIC SECTION

In the dynamic section all calculations are performed for each timestep of the model run. At the beginning of the section input time series (multi column ASCII files) are linked to maps to construct dynamic maps of these input time series.

```
###########################################################################
# The Dynamic model #######################################################
###########################################################################
dynamic

# Link timeseries to maps..
WindSpeedStation=timeinputscalar(WindSpeedTss,RainAreas);     #time series windspeed
WindDirStation=timeinputdirectional(WindDirTss,RainAreas);    #time series winddir
WindDirStationPoint=timeinputdirectional(WindDirTss,MeteoStations);    #time series
         winddir per stat
Precipitation=timeinputscalar(RainTss,RainAreas);      #time series precipitation

# Input temp and correct for altitude
Temperature=timeinputscalar(TempTss,RainAreas)+Alt2Temp;

# Time series  of  RH. No altitude correction Yet
RH = timeinputscalar(RHTss,RainAreas); # * RHCorr;
ShortWave=timeinputscalar(ShortWaveTss,RainAreas);
ShortWave=if(ShortWave < 0 then 0.0 else ShortWave);
HP=timeinputscalar(HPTss,RainAreas); # Horizontal component Juvik
Day=timeinputscalar(DayTime,1); # get julian day
Hour=timeinputscalar(DayTime,2); # get hour
```

USE TOPEX MAP TO DETERMINE WIND-SPEED AND -DIRECTION FOR EACH CELL

Here the topex maps are used to determine windspeed and direction for each cell of the area. The procedure needs adjusting if one or more of the gauges fall outside the catchment area. Start the process by determining the average wind direction.

```
###########################################################################
# Use Topex maps to determine wind-direction and speed for each cell #####
###########################################################################
# Determine average wind direction

Ux = maptotal(sin(WindDirStationPoint));
Uy = maptotal(cos(WindDirStationPoint));
# The above migth be a problem when using the -m option and if the
# actual gauges are outside the area. In that case use the lines below
# although there migth be overflow problems.
#Ux = maptotal(sin(WindDirStation));
#Uy = maptotal(cos(WindDirStation));

avwinddir = scalar(atan(Uy/Ux));
```

Check from which of the eight main directions the wind comes and select the appropriate map:

```
topex = if(avwinddir < 22.5 and avwinddir >= 0 then topexn else
         if(avwinddir < 67.5 and avwinddir >= 22.5  then topexne else
         if(avwinddir < 112.5 and avwinddir >= 67.5  then topexe else
         if(avwinddir < 157.5 and avwinddir >= 112.5  then topexse else
         if(avwinddir < 202.5 and avwinddir >= 157.5  then topexs else
         if(avwinddir < 247.5 and avwinddir >= 202.5  then topexsw else
         if(avwinddir < 292.5 and avwinddir >= 247.5  then topexw else
         if(avwinddir < 337.5 and avwinddir >= 292.5  then topexnw else

         if(avwinddir < 360.0 and avwinddir >= 337.5  then topexnw )
         ))))))));
```

Determine the relative exposure from the topex map:

```
ExposureWindSpdRatio=-0.005*topex+1.0; #Ruel had intercept of 1.5, -ive topex values are
        highly exposed

# Now get the ExposureWindSpdRatio for the stations... The problem is that some stations
        are outside the area.
StatExposurePoints = if(boolean(MeteoStations) then ExposureWindSpdRatio);
# make this an area..
StatExposure = areaaverage(StatExposurePoints, RainAreas);

# Hack to get relation between U and exposure for stations outside the area.
# Does not work for statiosn outside the area with the -m option
ExRel = areaaverage(WindSpeedStation/StatExposurePoints,TopoId);

# Now use this to fill in the missing station data
StatExposure = cover(StatExposure,WindSpeedStation / ExRel);

# The above migth fail, therefore: Last resort, fill with average..
StatExposure = cover(StatExposure,areaaverage(ExposureWindSpdRatio,TopoId));
```

Correct the exposure using the station exposure:

```
# Now use the difference in exposure to correct the exposure.
CorrectedExposureWindSpdRatio = ExposureWindSpdRatio/StatExposure;
# Finally correct the windspeed values
WindSpeed=WindSpeedStation*CorrectedExposureWindSpdRatio;

CumCorrectedExposureWindSpdRatio = CumCorrectedExposureWindSpdRatio
        +CorrectedExposureWindSpdRatio;
CumExposureWindSpdRatio = CumExposureWindSpdRatio + ExposureWindSpdRatio;
CumWindSpeed = CumWindSpeed + WindSpeed;
CumWindSpeedStation = CumWindSpeedStation + WindSpeedStation;

# At the moment We assume that the average wind-direction from the stations
# resembles the global direction ...
windldd = ldd(if(avwinddir < 22.5 and avwinddir >= 0 then windn else
        if(avwinddir < 67.5 and avwinddir >= 22.5  then windne else
        if(avwinddir < 112.5 and avwinddir >= 67.5  then winde else
        if(avwinddir < 157.5 and avwinddir >= 112.5  then windse else
        if(avwinddir < 202.5 and avwinddir >= 157.5  then winds else
        if(avwinddir < 247.5 and avwinddir >= 202.5  then windsw else
        if(avwinddir < 292.5 and avwinddir >= 247.5  then windw else
        if(avwinddir < 337.5 and avwinddir >= 292.5  then windnw else

        if(avwinddir < 360.0 and avwinddir >= 337.5  then windnw )
        )))))))));
# Get the wind-direction in degrees...
WindDir = scalar(directional(windldd));
```

DETERMINING AREAL AVERAGE PRECIPITATION

The next step in the dynamic section step is to correct the gauge information for wind losses. This function has been derived for the gauges used in the Rio Chiquito. This is a site/gauge specific function that should be re-determined when applying the model to another catchment. However, first the angle of P is estimated using the uncorrected data:

```
###########################################################################
# First correct the precipitation for Windloss ##########################
###########################################################################
# Windloss function for Costa Rica By Frumau but limited to 4m/s
# Use the STATION windspeed to correct the P input. Done for the whole grid but
# used for the station points only.

# First estimation of P angle using uncorrected P
Ratio = if (Precipitation > 0.0 then HP/Precipitation else 0.0);
```

```
HPPrecipAngle = if (Precipitation == 0.0 then
                    if (HP > 0.0 then 90 else 0.0)
                else scalar(atan(Ratio)));

# Correct precipitation amount. CQ Specific function!
PrecPol = Precipitation;
CumPrecPol = CumPrecPol + PrecPol; # original P
PCorrFac =  0.01 * exp(4.606 - 4.4406E-7 * HPPrecipAngle**3.166);
Precipitation = Precipitation / PCorrFac;
PrecWindCor=Precipitation;
CumPrecWindCor=CumPrecWindCor+PrecWindCor;

# Correct for measurement errors. CQ Specific function!
HP = HP * 0.9;
# Now correct the HP component, the the is catching efficieny of the juvic
# gauge.
HPGaugeEff = min(1, 0.36 * 0.5 * HP + 0.2); #CQ Specific function!
HP = HP/HPGaugeEff;
```

Next the angle (in degrees from the vertical) of the incoming precipitation is determined using both components of the Juvic gauge. Thhis time we use the corrected P and HP:

```
##########################################################################
# Determine precipitation angle ##########################################
##########################################################################

##########################################################################
# Again Determine precipitation angle using Juvik gauge info #############
# but now with corrected P  ##############################################
##########################################################################

Ratio = if (Precipitation > 0.0 then HP/Precipitation else 0.0);
HPPrecipAngle = if (Precipitation == 0.0 then
                    if (HP > 0.0 then 90 else 0.0)
                else scalar(atan(Ratio)));

CumHPPrecAngle = CumHPPrecAngle + HPPrecipAngle;
```

Alternatively a regression between wind speed and rainfall intensity (after wind correction) may be used to determine the angle. This option is commented out in the current code:

```
#b = ln(2+0.38 * Precipitation ** 0.42);
#a = 1/ln(e + 10.46 * Precipitation ** 1.24);
#PrecipAngle=90- exp(4.5 - a * WindSpeed ** b);
```

Now that we have the angle at the gauges we can determine the angle at each cell in the model. First the angle at the gauge is used to correct the rainfall input. With this angle and the vertically measured precipitation the precipitation on a plane at a 90 degrees angle to the precipitation vector is determined. This amount is assumed to be the potential real precipitation.

```
##########################################################################
# Determine P angle for each cell in the area and ########################
# use the P angle to correct hydrological input ##########################
##########################################################################
# With the angle and the vertically measured precipitation determine the
# precipitation on a plane at a 90deg angle to the precipitation vector. This
# is the assumed potential real precipitation. This is assumed to be identical for
# each cell in the area.
# We used to do this from P but max of HP and P is better....

PotPrecipitation_P=Precipitation/scalar(cos(HPPrecipAngle));
PotPrecipitation_HP=if(HPPrecipAngle > 0 then HP/scalar(sin(HPPrecipAngle)) else 0.0);

PotPrecipitation=max(PotPrecipitation_P,PotPrecipitation_HP);

CumPotPrec = CumPotPrec + PotPrecipitation;

# Keep track of HP only events. For those events the Windspeed function
# is not valid and we need to make sure the angle is 90degres: fog only.
```

```
# For the for only events we need to scale according to the windspeed
# ratio between gauge and actual cell as the catching in determined
# but the windspeed (flux). Afterwards multiply by catcheff

Fogonly = if (HPPrecipAngle == 90 then 1 else 0);

# Now use the regression between Potential P and windspeed to
# get P angle for each cell in the area. Use Pp because Arnouds
# regression is based on 30 min values...
# CQ Specific function!
Pp = PotPrecipitation/4.0;
a = -0.0001477 *Pp**3 + 0.00686*Pp**2 - 0.104*Pp + 0.621;
b = 8.803E-05*Pp**3 - 0.00502*Pp**2 + 0.0871*Pp + 0.659;
b = if(Pp > 20 then 1.0972 else b);
a = if(Pp > 20 then 0.1034 else a);
PrecipAngle= 90 - exp(4.5 - a * WindSpeed ** b);

# Force Fog-only events over the area for a gauge...
PrecipAngle = if (Fogonly then 90 else PrecipAngle);

CumPrecAngle = CumPrecAngle +PrecipAngle;
```

Subsequently, the wind direction and angle are used in combination with the aspect to determine the angle between the surface and the precipitation to correct the precipitation input per cell. The angle of the terrain has been determined from the DEM in the initial section of the program. The horizontal precipitation angle is determined from the wind direction for each time step using the Juvik gauge input.

```
# Now use winddirection and angle in combination with the aspect to the angle
# between slope and precipitation. This results in a corrected input in mm.
# The first step is to determine the angle of the terrain from the slope. This has
# been done in the initial section)

# Horizontal precipitation angle determined by the winddir...
PrecipAspect=scalar(WindDir);
```

Now calculate the input angle:

```
# Now calculate the input angle according to:
Tmp = max(-1,min(1,cos(Terrain_angle)*cos(PrecipAngle)+sin(Terrain_angle)*
         sin(PrecipAngle)*cos(scalar(Aspect)-scalar(PrecipAspect))));
Input_angle=scalar(acos(Tmp));
```

Remove angles > 90 degrees as this represent precipitation going through the ground.

```
Input_angle = max (0, min(90, Input_angle));
```

Now we can calculate the amount of rain on the sloping surface.

```
Precipitation=PotPrecipitation*scalar(cos(Input_angle));
```

Because in a hydrological model the rest of the fluxes are calculated on a horizontal surface convert the amount back to a horizontal surface.

```
Precipitation=Precipitation/scalar(cos(Terrain_angle));
```

The following figures show the effect of applying these corrections.

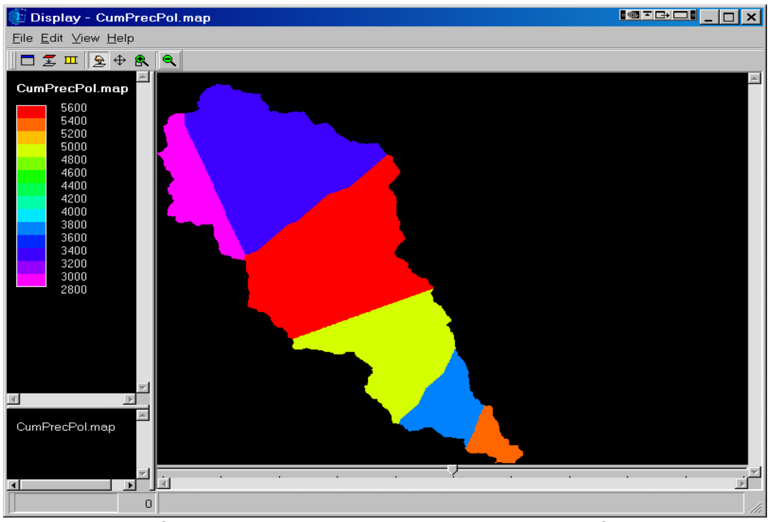


Illustration 4 Cumulative precipitation over the Rio Chiquito Catchment determined using gauge information and Thiesen Polygons

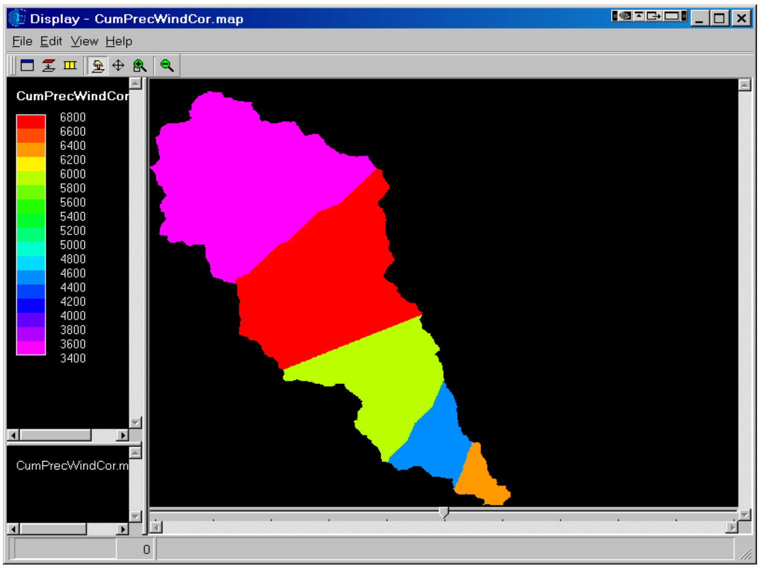


Illustration 5 Cumulative precipitation over the Rio Chiquito Catchment after applying the wind-speed corrections.

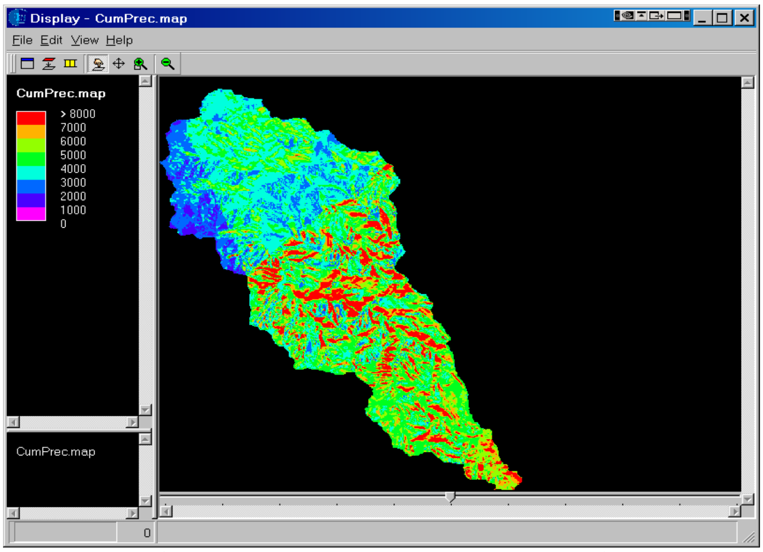


Illustration 6 Cumulative precipitation over the Rio Chiquito Catchment after applying the corrections for precipitation Angle and aspect

The following table summarizes the effects of the different precipitation corrections over the entire Rio Chiquito catchment. The averages have been determined for the catchment part of the DEM only using the July 2003 to July 2004 data:

	P polygon	***P Wind correction***	***P after Angle/Aspect correction***
Amount [mm/yr]	4309	5087	4681
% of Final	92,05	108,67	100

Fog interception (HP)

For interception is calculated by two methods. For event that are fog only (determined by the fact that only the horizontal component of the Juvik gauge holds water, i.e. for angles of 90 degrees) and for events in which the incoming angle is > 85 degrees and < 90 degrees. At the end of the calculations the estimated amount of fog interception is added to the canopy storage.

```
###########################################################################
# Fog interception framework##############################################
###########################################################################

# Estimate Juvik hori input for all other cells using Potential P and
# the rain angle. Force HP to be the gauge value for Fog only events.
# Correct HP for fog only event using cell windspeed compared to
# gauge windspeed.
# Pot = HP/sin(angle)
HP = if (Fogonly then HP * (max(1,WindSpeed))/(max(1,WindSpeedStation)) else
         PotPrecipitation * sin(PrecipAngle));

Precipitation = if(PrecipAngle > CriticalAngle then 0 else Precipitation);
Precipitation = if(Fogonly then 0 else Precipitation);
```

```
# Set to have Fog if the vertical component  is very small otherwise
# all is wind-driven rain.
JuvikFog = if (PrecipAngle > CriticalAngle then HP else 0.0);
JuvikFog = if(Fogonly then HP else JuvikFog);
# Now apply the Vegetation Catching efficiency
WaterCatch = JuvikFog * CatchEff;
CumWaterCatch = CumWaterCatch + WaterCatch;

# Add the Water that has been catched to the Canopystorage;
CanopyStorage = CanopyStorage + WaterCatch;
```

DETERMINE RADIATION INPUT

Actual radiation input per cell is determined in a similar wat to precipitation input. However in this case shading due to peaks in the landscape is taken into account using the horizontan function in `pcraster`.

First the solar geometry is determined using location information and the time and day of the year.

```
##########################################################################
# Calculate Solar Angle and correct radiation ###########################
##########################################################################
# Radiation model by Oscar NB only works between -23 and + 23 degrees!!
#=========================================================
##########################################################
# Daily calculation loop all models
# ---------------------------

# Solar geometry
# ----------------------------
# SolDec  :declination sun per day  between +23 & -23 [deg]
# HourAng :hour angle [-] of sun during day
# SolAlt  :solar altitude [deg], height of sun above horizon
# SolDec  = -23.4*cos(360*(Day+10)/365);
# Now added a new function that should work on all latitudes!
 theta    =(Day-1)*360/365;  # day expressed in degrees

 SolDec =180/pi * (0.006918-0.399912 * cos(theta)+0.070257 * sin(theta) -   0.006758 *
        cos(2*theta)+0.000907 * sin(2*theta) -  0.002697 * cos(3*theta)+0.001480 *
        sin(3*theta));

 HourAng = 15*(Hour-12.01);
 SolAlt  = scalar(asin(scalar(sin(Latitude)*sin(SolDec)+cos(Latitude)*
          cos(SolDec)*cos(HourAng))));


# Solar azimuth
# ----------------------------
# SolAzi  :angle solar beams to N-S axes earth [deg]
 SolAzi = scalar(acos((sin(SolDec)*cos(Latitude)-cos(SolDec)*
         sin(Latitude)*cos(HourAng))/cos(SolAlt)));
 SolAzi = if(Hour le 12 then SolAzi else 360 - SolAzi);

# Surface azimuth
# ----------------------------
# cosIncident :cosine of angle of incident; angle solar beams to angle surface
 cosIncident = sin(SolAlt)*cos(Slope)+cos(SolAlt)*sin(Slope)
              *cos(SolAzi-Aspect);
# report cosi.tss = timeoutput(Loc,cosIncident);
```

The critical sun angle determines at what solar angles a cell in the DEM receives radiation.

```
# Critical angle sun
# ----------------------------
# HoriAng  :tan maximum angle over DEM in direction sun, 0 if neg
# CritSun  :tan of maximum angle in direction solar beams
# Shade    :cell in sun 1, in shade 0
# NOTE: for a changing DEM in time use following 3 statements and put a #
```

```
#       for the 4th CritSun statement

 HoriAng   = horizontan(Altitude,directional(SolAzi));
 HoriAng   = if(HoriAng lt 0 then scalar(0) else HoriAng);
 CritSun   = if(SolAlt gt 90 then scalar(0) else scalar(atan(HoriAng)));
 Shade   = SolAlt gt CritSun;

# Radiation outer atmosphere
# ----------------------------
 OpCorr = Trans**((sqrt(1229+(614*sin(SolAlt))**2)
         -614*sin(SolAlt))*AtmPcor);    # correction for air masses [-]
 Sout   = Sc*(1+0.034*cos(360*Day/365)); # radiation outer atmosphere [W/m2]
 Snor   = Sout*OpCorr;                  # rad on surface normal to the beam [W/m2]
# Radiation at DEM
# ----------------------------
# Sdir   :direct sunlight on a horizontal surface [W/m2] if no shade
# Sdiff  :diffuse light [W/m2] for shade and no shade
# Stot   :total incomming light Sdir+Sdiff [W/m2] at Hour
# Radiation :avg of Stot(Hour) and Stot(Hour-HourStep)
# NOTE: PradM only valid for HourStep & DayStep = 1
 SdirCor   = if(Snor*cosIncident*scalar(Shade)<0,0.0,Snor*cosIncident*scalar(Shade));
 Sdir   = if(Snor*cosIncident<0,0.0,Snor*cosIncident);
 Sdiff  = if(Sout*(0.271-0.294*OpCorr)*sin(SolAlt)<0, 0.0,
         Sout*(0.271-0.294*OpCorr)*sin(SolAlt));
AtmosDiffFrac = if(Sdir > 0 then Sdiff/Sdir else 1);

# Stot   = cover(Sdir+Sdiff,windowaverage(Sdir+Sdiff,3));                        # Rad
        [W/m2]
 Stot   = Sdir + Sdiff;                                                          # Rad
        [W/m2]
 StotCor   = SdirCor + Sdiff;                                             # Rad [W/m2]
 PotenRad = (Sold + Stot)/2;
                  # Rad interval
 PotenRadCor = (SoldCor + StotCor)/2;                                     # Rad interval

 Sold = Stot;
 SoldCor = StotCor;
```

Now use the measured incoming radiation and correct this using the solar angle information. For that we also need to estimate the amount of diffusive radiation.

```
# Estimate diffusive radiation from measured short wave and possible incoming radiation
DiffRadFrac = max(0.1,min(1,1.0-if(PotenRad<=0 then 1 else (ShortWave/PotenRad))));

# Make sure that the estimate is not smaller than the atmospheric diffusive radiation
        fraction
# Convert measured shortwave radiation to radiation on surface normal to beam
# - correct only if the direct part > 50 %
# - if so only correct the direct part
# - also make sure there are no problems at very low solar angles
#   by not allowing the dividing factor to be smaller than 0.2
CorRad = if (DiffRadFrac < 0.5 then
                           (1- DiffRadFrac) * ShortWave/max(0.2,scalar(cos(90 - SolAlt)))
        + (DiffRadFrac * ShortWave)
                                                            else
                           ShortWave);

# Direct Radiation at the surface
CorRadDir = (1-DiffRadFrac) * if(CorRad*cosIncident*scalar(Shade) < 0.0 then 0.0 else
        CorRad*cosIncident*scalar(Shade));

# Diffuse Radiation at the surface
CorRadDiff = max(0,DiffRadFrac * CorRad);
```

```
CorRad = CorRadDiff + CorRadDir;

# make sure no missing points exist....
CorRad=cover((CorRad*1),(Altitude * 0) + areaaverage(CorRad*1,boolean(Altitude))); #
        hourly rad [W/m2]

CumRad = CorRad + CumRad;

Radiation = CorRad;
```

The following figure shows the average radiation over a whole year determine using the method described above.

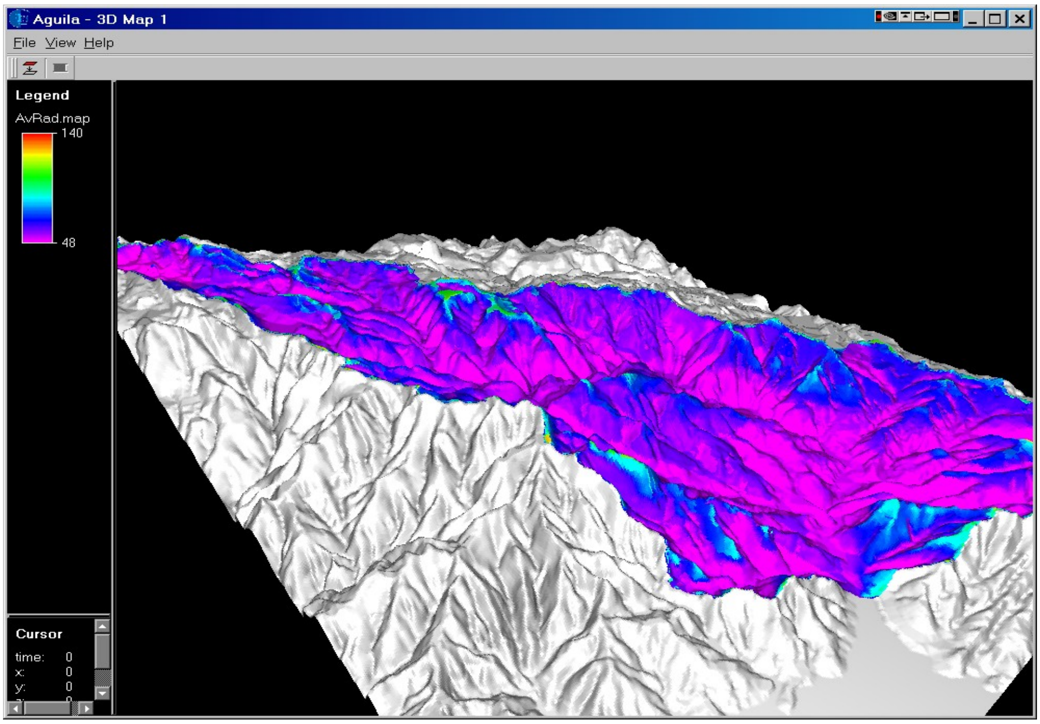


Illustration 7 Average radiation [w/m^2] over the whole modelling period

EVAPORATION

Evaporation (for both the wet and dry canopy) is calculated using using the Penman-Monteith formula. If the canopy is (partly) wet all energy is used fro wet canopy evaporation. If all water has evaporated in a time step the left-over energy is used fro transpiration. A fraction of the radiation is diverted to the soil surface and used for soil evaporation.

```
###########################################################################
# Calculate Penman Montieth evaporation ##################################
###########################################################################

# Estimate Soil Heat Flux from radiation and leaf area index
G = Radiation * (1- Albedo) * 1/(LeafAreaIndex+12);

# Determine Esat and Delta according to Calder 1990
Esat = 6.1078*exp(17.2694*Temperature/(Temperature+237.3));
Delta = Esat*17.2694*237.3/sqr(Temperature+237.3);

# Determine Eact using relative humidity
Eact = RH * Esat / 100;

# Determine specific heat of air
Lambda =   4185.5 * (751.78 - (0.5655 * (Temperature + 273.15)));
```

```
# Now determine Gamma
p = 900.0;        # pressure in mb
cp = 1005.0;      # J/(kgK)
Gamma =   (cp * p)/(0.622 * Lambda);  #  mbC
# density of dry air in kg/m^3
rho = 1.201 * (290 * ( p - 0.378 * Eact)/(1000 * (Temperature + 273.15)));
#rho = hml_rho(p,Eact,Temperature);


# At present set A to Radiation * (1- Albedo) and split according to
# wetted part of the Canopy
WetPart = min(1,CanopyStorage/Cmax);
Atrans = (Radiation -G) * (1 - Albedo) * (1 - WetPart);
A = (Radiation - G) * (1 - Albedo);
Acanopy = (Radiation - G) * (1 - Albedo);
Aint = Acanopy - Atrans;

# Potential in mm temp var, needed for check
AtransMM = Atrans/(Delta+Gamma)/Lambda;
AintMM = Aint/(Delta+Gamma)/Lambda;


# Determine Ra as a function of Windspeed and canopy parameters
# Calculates ra (aerodynamic resistance) according to Arnouds function
# CQ Specific function!
z = WindSpeedHeigth;
Zom = VegetationHeigth * 0.123;
Zoh = 0.25 * Zom;
d = 0.66 * VegetationHeigth;

Ra = 4.72 * ln((z-d)/Zom) * ln((z-d)/Zoh)/(1 + 0.54 * WindSpeed);


# Now the actual formula, this is for Interception, rs is zero
VPD = (Esat - Eact);
n = Delta + Gamma; # for interception Rs = 0;

tmp = Rs/Ra;
nn = Delta + (Gamma * (1 + tmp));

t = (Delta * A) + (rho * cp * VPD / Ra);
#t = (Delta * Aint) + (rho * cp * VPD / Ra);
EA = (t / n);

PotEvap = EA/Lambda * timestepsecs; # now in mm

# Now the actual formula, this is for Transpiration
# Determine Rs seperate for Pasture and Forest (Hard coded, should be paramiterized in
         files later)
# Reference equations
InVPD = if(VPD < 0.01 then 0.01 else VPD);
InWave = if(ShortWave < 5 then 5 else ShortWave);
# CQ Specific function!
PasRs=exp(1.05 * ln(InVPD) - 0.651 * ln(InWave) + 5.89);
# CQ Specific function!
ForRs=exp(0.867 * ln(InVPD) - 0.000831 * InWave + 2.81);

Rs = max(0.5,min(1000,if (scalar(LandUse) > 1 then ForRs else PasRs)));

# No transpiration at nigth, this is of no use to the trees.
Rs = if (InWave < 10.0 then 500 else Rs);

# Soil evaporation is disabled
# Now the actual formula, this is for the soil, rs is zero
# use vegetation Ra
#t = (Delta * Asoil) + (rho * cp * (VPD) / Ra);
```

```
#tmp = Rs/Ra;
#n = Delta + (Gamma * (1 + tmp));

#Erad = (Delta * Asoil) / (Delta + Gamma * (1 + tmp));
#Ea = rho * cp * (VPD)/Ra;
#Ewind = Ea/(Delta + Gamma * (1 + tmp));

#EA = (t / n);
#SoilEvap = EA/Lambda * timestepsecs; # now in mm
```

INTERCEPTION

The interception model is based on a simplified version of the Rutter model. It used hourly timesteps.

```
###########################################################################
# Interception according to a modified Rutter model with hourly timesteps#
###########################################################################

p = CanopyGapFraction;
pt = 0.1 * p;

# Amount of P that falls on the canopy
Pfrac = (1 - p -pt) * Precipitation;

# S cannot be larger than Cmax, no gravity drainage bolow that
DD = if (CanopyStorage > Cmax then Cmax - CanopyStorage else 0.0);
CanopyStorage = CanopyStorage - DD;

# Add the precipitation that falls on the canopy to the store
CanopyStorage = CanopyStorage + Pfrac;

# Now do the Evap, make sure the store does not get negative
dC = -1 * min(CanopyStorage, PotEvap);
CanopyStorage = CanopyStorage + dC;

LeftOver = PotEvap +dC; # Amount of evap not used


# Now drain the canopy storage again if needed...
D = if (CanopyStorage > Cmax then CanopyStorage - Cmax else 0.0);
CanopyStorage = CanopyStorage - D;

# Calculate throughfall
ThroughFall = DD + D + p * Precipitation;
StemFlow = Precipitation * pt;

# Calculate interception, this is NET Interception
NetInterception = Precipitation - ThroughFall - StemFlow;
Interception = -dC;

# Determine Evnergy left over for transpiration
Atrans = if (CanopyStorage > 0.001 then if(PotEvap > 0 then LeftOver/PotEvap * A else A)
         else A);

t = (Delta * Atrans) + (rho * cp * VPD/ Ra);
#t = (Delta * A) + (rho * cp * VPD/ Ra);
tmp = Rs/Ra;
n = Delta + (Gamma * (1 + tmp));

EA = (t / n);

PotTrans = EA/Lambda * timestepsecs; # now in mm


RestPotEvap= PotTrans;
```

SOIL WATER BALANCE

The soil model is based on the TOPO_SBM concept. As a first step the transpiration is extracted from the soil. First from the unsaturated zone scaling the amount using the depth of the unsaturated zone in relation to the rootingdepth of the vegetation. Any leftover transpiration is taken from the saturated part of the soil if the roots can get to that part.

```
###########################################################################
# Start with the soil calculations  #####################################
###########################################################################

# Distributed soil model with two layers
# Possible improvements:
# - add thetaS and thetaR to get actual water table depth and soil depth. Now all
#   is in water capacity, not real depth



ExfiltWater=ZeroMap;
FreeWaterDepth=ZeroMap;

ExfiltWater=if (FirstZoneDepth - FirstZoneCapacity > 0 then FirstZoneDepth -
         FirstZoneCapacity else 0.0);
FirstZoneDepth=FirstZoneDepth - ExfiltWater;


###########################################################################
# Evapotranspiration in SBM model ########################################
###########################################################################
# Step 1 try from unsaturated store...
# First decrease the amount of available water from the unsaturated
# zone according to the rooting depth and the depth of the water table
PotenEvap = RestPotEvap;
AvailCap = if(zi < RootingDepth then
                zi/RootingDepth
           else
                 RootingDepth/zi);
MaxExtr = AvailCap  * UStoreDepth*(1-thetaR);
ActEvapUStore = if ( MaxExtr < RestPotEvap then MaxExtr else RestPotEvap);
UStoreDepth = UStoreDepth - ActEvapUStore;
RestPotEvap = RestPotEvap - ActEvapUStore;

# Step 2 do rest from saturated zone, use rootingDepth as a limiting factor
ActEvapSat = min( if(zi > RootingDepth then
                                    0.0
                           else
                           FirstZoneDepth*(1-thetaR)),RestPotEvap);

FirstZoneDepth = FirstZoneDepth - ActEvapSat;
ActEvap = ActEvapSat + ActEvapUStore;
```

The amount of water infiltrating into the soil is determined. All infiltrating water enters the unsaturated store first. The model allows part of a cell to be compacted and have a different infiltration capacity.

```
###########################################################################
# Determine infiltration into Unsaturated store...########################
###########################################################################
# Add precipitation surplus to FreeWater storage...

FreeWaterDepth=ThroughFall + StemFlow;
UStoreCapacity = FirstZoneCapacity - FirstZoneDepth - UStoreDepth;

# First determine if the soil infiltration capacity can deal with the
# amount of water
# split between infiltration in undisturbed soil and compacted areas (paths)
```

```
#OldUStoreCapacity = UStoreCapacity;
MaxInfiltPath= min(InfiltCapPath,FreeWaterDepth);
PathInfiltExceeded=PathInfiltExceeded + if(InfiltCapPath < FreeWaterDepth then 1 else 0);
InfiltPath = PathFrac * min(MaxInfiltPath, UStoreCapacity);
UStoreDepth = UStoreDepth + InfiltPath;
UStoreCapacity = UStoreCapacity - InfiltPath;
FreeWaterDepth = FreeWaterDepth - InfiltPath;
# report e:\temp\AA_FW=FreeWaterDepth;

# Now increase the amount of water that needs to infiltrate in the soil with
# infiltration excess water from the paths
# Assume that everthing left MUST infiltrate into the soil part

MaxInfiltSoil= min(InfiltCapSoil,FreeWaterDepth/(1-PathFrac));
SoilInfiltExceeded=SoilInfiltExceeded + if(InfiltCapSoil < FreeWaterDepth/(1- PathFrac)
         then 1 else 0);
InfiltSoil = (1 - PathFrac) * min(MaxInfiltSoil, UStoreCapacity);
UStoreDepth = UStoreDepth + InfiltSoil;
UStoreCapacity = UStoreCapacity - InfiltSoil;
FreeWaterDepth = FreeWaterDepth - InfiltSoil;
# report e:\temp\A_FW=FreeWaterDepth;
ActInfilt = InfiltPath + InfiltSoil;

InfiltExcess = if (UStoreCapacity > 0 then FreeWaterDepth else 0);
CumInfiltExcess=CumInfiltExcess+InfiltExcess;
```

Next, water is transferred from the unsaturated store to the saturated store. The transfer is based on the saturated conductivity at the depth of the (pseudo) water table. For the special case of the Rio Chiquito a leakage term has been introduced using an exponential function.

```
###########################################################################
# Transfer of water from unsaturated to saturated store...################
###########################################################################
zi = FirstZoneCapacity - FirstZoneDepth;
                  #/(thetaS -thetaR); # Determine actual water depth
Ksat = FirstZoneKsatVer * exp(-f * zi);

# Determine saturation deficit
SaturationDeficit =  FirstZoneCapacity - FirstZoneDepth;

# now the actual tranfer to the saturated store..
Transfer = min(UStoreDepth*(1-thetaR),if (SaturationDeficit <= 0.00001 then 0.0 else Ksat
         *
UStoreDepth/(SaturationDeficit+1)));

# Now add leakage
# Limit to MaxLeakage/day. Leakage percentage gets bigger if the
# storm is bigger (macropores start kicking in...
ActLeakage = max(0,min(MaxLeakage/24,Transfer * exp(0.01 * Transfer)/e));
Transfer = Transfer - ActLeakage;


FirstZoneDepth = FirstZoneDepth + Transfer;
#report $2\AA_FD=FirstZoneDepth;
#report $2\AA_UC=UStoreCapacity;
#report $2\AA_UD=UStoreDepth;

UStoreDepth = UStoreDepth - Transfer - ActLeakage;

# Determine % saturated
Sat = FirstZoneDepth >= (FirstZoneCapacity*0.999);
PercSat = areaaverage(scalar(Sat),TopoId) * 100;
```

Horizontal transport of water is performed using the accucapacityflux functionality. The amount of water to transfer downstream is determined using the slope of each cell in combination with the saturated conductivity and the saturation deficit of the cell. The flux is limited to the available amount of water in a cell.

```
###########################################################################
```

```
# Horizontal (downstream) transport of water ############################
########################################################################
#MaxHor = FirstZoneKsatVer * tan (Beta) * exp(-SaturationDeficit/M);
MaxHor = max(0,min(FirstZoneKsatVer * Slope * exp(-
         SaturationDeficit/M),FirstZoneDepth*(1-thetaR)));
FirstZoneFlux,FirstZoneDepth = accucapacityflux,accucapacitystate (TopoLdd,
         FirstZoneDepth, MaxHor);
```

After the horizontal transport all cells that have more water than they can store will produce returnfllow (water exfiltrating from the soil). This water together with other surface water (infiltration excess water) is converted to m^3/sec for use in the kinematic wave function.

```
########################################################################
# Determine returnflow from first zone          ##########################
########################################################################
ExfiltWater=ExfiltWater + if (FirstZoneDepth - FirstZoneCapacity > 0 then FirstZoneDepth
         - FirstZoneCapacity else 0.0);
FirstZoneDepth=FirstZoneDepth - ExfiltWater;
# Added excess UStore water to exfiltrated water
OldUStoreDepth = UStoreDepth;
UStoreDepth = if (ExfiltWater > 0 then 0 else UStoreDepth);
ExFiltWater = ExfiltWater + OldUStoreDepth - UStoreDepth;

# Re-determine UStoreCapacity
UStoreCapacity = FirstZoneCapacity - FirstZoneDepth - UStoreDepth;

# report e:\temp\B_FD=FirstZoneDepth;
# report e:\temp\B_UC=UStoreCapacity;
# report e:\temp\B_UD=UStoreDepth;

# SurfaceMM
SurfaceWater = SurfaceRunoff * DCL * timestepsecs/QMMConv;
CumSurfaceWater = CumSurfaceWater + SurfaceWater;
# Determine surface water that should be routed downstream using a kinematic wave
#Inwater=((ExfiltWater + FreeWaterDepth) * reallength * reallength * 0.001) /
         timestepsecs;

# Estimate water that may re-infiltrate
Reinfilt = max(0,min(SurfaceWater+FreeWaterDepth,min(InfiltCapSoil,UStoreCapacity)));
CumReinfilt=CumReinfilt+Reinfilt;
Inwater=((ExfiltWater + FreeWaterDepth-Reinfilt) * reallength * reallength * 0.001) /
         timestepsecs;

UStoreDepth = UStoreDepth + Reinfilt;
```

All surface water is used as input for the kinematic wave function. This water is stored in a separate store that does not allow re-infiltration of water.

```
########################################################################
# Runoff calculation via Kinematic wave ################################
########################################################################
# per distance along stream ((m3/s)/m))
q=Inwater/DCL;

# discharge (m3/s)
SurfaceRunoff = kinematic(TopoLdd, SurfaceRunoff,q,Alpha, Beta,timestepsecs,reallength);
SurfaceRunoffMM=SurfaceRunoff *  timestepsecs/QMMConv;

# water depth (m)
H=(Alpha*(SurfaceRunoff**Beta))/Bw;
# wetted perimeter (m)
P=Bw+2*H;
# Alpha
```

```
Alpha=AlpTerm*(P**AlpPow);

Runoff=SurfaceRunoff;
```

The rest of the program deals with reporting and water budget determination only. Extra reporting can be done by adding statements or removing comments from certain lines. See the section in the user manual on adjusting the program.

```

##########################################################################
# water balance (not complete!)##########################################
##########################################################################

# Catchment water budget
sumprecip=sumprecip+areaaverage(Precipitation,TopoId);
sumevap=sumevap+areaaverage(ActEvap,TopoId);
sumrunoff=1;#sumrunoff+scalar(areaaverage(SurfaceRunoffMM,OutputLoc));
sumint=sumint+areaaverage(Interception,TopoId);
sumoutflow=1;#sumoutflow+areaaverage(FirstZoneFlux,OutputLoc);
nowstorage=areaaverage(FirstZoneDepth,TopoId)+areaaverage(UStoreDepth,TopoId)+areaaverage(
         CanopyStorage,TopoId)+areaaverage(SurfaceRunoffMM,TopoId);
sumleakage=sumleakage+areaaverage(ActLeakage,TopoId);

#water balance: watbal should always be equal to zero (or 10**-5 for computer errors)
watbal=sumprecip-sumevap-sumrunoff-sumint-sumleakage-sumoutflow+(initstorage-nowstorage);

# Single cell based water budget
OldCellStorage = CellStorage;
CellStorage = UStoreDepth+FirstZoneDepth;
OutFlow = FirstZoneFlux;
InFlow = upstream(TopoLdd,FirstZoneFlux);
CellWatBal = ActInfilt - ActEvap - ExfiltWater - ActLeakage + Reinfilt + InFlow - OutFlow
         + (OldCellStorage - CellStorage);
SumCellWatBal = SumCellWatBal + CellWatBal;


CumPrec=CumPrec+Precipitation;
CumEvap=CumEvap+ActEvap;
CumPotenEvap=CumPotenEvap+PotenEvap;
CumInt=CumInt+Interception;
CumLeakage=CumLeakage+ActLeakage;

####################################################################
############ Below is for reporting only ###########################
####################################################################

CumExfiltWater=CumExfiltWater+ExfiltWater;


# Time series reporting
report $4\PercSat.tss=timeoutput(OutputId,PercSat);#{
#PercSat: Percentage of catchment that is saturated }
report $4\DischargeMM.tss=timeoutput(OutputLoc,SurfaceRunoffMM);

report $4\Exmm.tss=timeoutput(OutputId,areaaverage(ExfiltWater,TopoId));
report $4\FreeWaterDepth.tss=timeoutput(OutputId,areaaverage(FreeWaterDepth,TopoId));
report $4\ActInfilt.tss=timeoutput(OutputId,areaaverage(ActInfilt,TopoId));
report $4\Reinfilt.tss=timeoutput(OutputId,areaaverage(Reinfilt,TopoId));
report $4\UStoreDepth.tss=timeoutput(OutputId,areaaverage(UStoreDepth,TopoId));
report $4\FirstZoneDepth.tss=timeoutput(OutputId,areaaverage(FirstZoneDepth,TopoId));
report $4\InfiltExcess.tss=timeoutput(OutputId,areaaverage(InfiltExcess,TopoId));
report $4\ActLeakage.tss=timeoutput(OutputId,areaaverage(ActLeakage,TopoId));
report $4\ActEvapSat.tss=timeoutput(OutputId,areaaverage(ActEvapSat,TopoId));
report $4\PotenEvap.tss=timeoutput(OutputId,areaaverage(PotenEvap,TopoId));
report $4\ActEvapUStore.tss=timeoutput(OutputId,areaaverage(ActEvapUStore,TopoId));

report $4\FF.tss=timeoutput(OutputId,areaaverage(FirstZoneFlux,TopoId));
report $4\rl.tss=timeoutput(OutputLoc,reallength);
report $4\HH.tss=timeoutput(OutputLoc,H);
report $4\QQ.tss=timeoutput(OutputLoc,SurfaceRunoff);
```

```
report $4\FZ.tss=timeoutput(OutputLoc,FreeWaterDepth);

report $4\WetPart.tss=timeoutput(OutputId,areaaverage(WetPart,TopoId));
report $4\PrecPol.tss=timeoutput(OutputId,areaaverage(PrecPol,TopoId));
report
        $4\ExposureWindSpdRatio.tss=timeoutput(OutputId,areaaverage(ExposureWindSpdRatio
        ,TopoId));
report
        $4\CorrectedExposureWindSpdRatio.tss=timeoutput(OutputId,areaaverage(CorrectedEx
        posureWindSpdRatio,TopoId));
report $4\PotPrec.tss=timeoutput(OutputId,areaaverage(PotPrecipitation,TopoId));
report $4\Transfer.tss=timeoutput(OutputId,areaaverage(Transfer,TopoId));
report $4\PrecWindCor.tss=timeoutput(OutputId,areaaverage(PrecWindCor,TopoId));
report $4\GroundHeatFlux.tss=timeoutput(OutputId,areaaverage(G,TopoId));

report $4\ShortWave.tss=timeoutput(OutputId,areaaverage(ShortWave,TopoId));
report $4\Radiation.tss=timeoutput(OutputId,areaaverage(Radiation,TopoId));
report $4\PotentialRadiationHOR.tss=timeoutput(OutputId,areaaverage(PotenRad,TopoId));
report
        $4\PotentialRadiationCatch.tss=timeoutput(OutputId,areaaverage(PotenRadCor,TopoI
        d));


avrunoff=areamaximum(SurfaceRunoff,TopoId);
report $4\DischargeCubic.tss=timeoutput(OutputLoc,SurfaceRunoff);


avevap=areaaverage(ActEvap,TopoId);                #evaporation
report $4\Evaporation.tss=timeoutput(OutputId,avevap);

avpotevap=areaaverage(PotEvap,TopoId);             #potential evaporation wet canopy
report $4\WetCanopyEvaporation.tss=timeoutput(OutputId,avpotevap);

pottrans=areaaverage(PotTrans,TopoId);             #potential transpiration
report $4\PotentialTranspiration.tss=timeoutput(OutputId,pottrans);

avprecip=areaaverage(Precipitation,TopoId);        #precipitation
report $4\Precipitation.tss=timeoutput(OutputId,avprecip);

avtf=areaaverage(ThroughFall,TopoId);        #Throughfall
report $4\Throughfall.tss=timeoutput(OutputId,avtf);

avc=areaaverage(CanopyStorage,TopoId);        #CanopyStorage
report $4\CanopyStorage.tss=timeoutput(OutputId,avc);

avd=areaaverage(D,TopoId);        #Drainage
report $4\CanopyDrainage.tss=timeoutput(OutputId,avd);

avint=areaaverage(Interception,TopoId);        #Drainage
report $4\Interception.tss=timeoutput(OutputId,avint);

avint=areaaverage(NetInterception,TopoId);        #Drainage
report $4\NetInterception.tss=timeoutput(OutputId,avint);

avwct=areaaverage(WaterCatch,TopoId);        #Water Catched by the canopy
report $4\WaterCatch.tss=timeoutput(OutputId,avwct);

aint=areaaverage(Aint,TopoId);        #Available energy interception
report $4\A_Interception.tss=timeoutput(OutputId,aint);

atrans=areaaverage(Atrans,TopoId);        #Available energy transpiration
report $4\A_Transpiration.tss=timeoutput(OutputId,atrans);

aa=areaaverage(A,TopoId);        #Available energy
report $4\A_All.tss=timeoutput(OutputId,aa);
```

```
#asoil=areaaverage(Asoil,TopoId);        #Available energy soil
#report $4\A_Soil.tss=timeoutput(OutputId,asoil);

acanopy=areaaverage(Acanopy,TopoId);        #Available energy canopy
report $4\A_Canopy.tss=timeoutput(OutputId,acanopy);


avgrad=areaaverage(Radiation,TopoId);                #PotRad
report $4\IncomingRadiation.tss=timeoutput(OutputId,avgrad);

report $4\WaterBalance.tss=timeoutput(OutputId,watbal);

report $4\ActualStorage.tss=timeoutput(OutputId,nowstorage);


# These report the variables per land use type
################################################
lu_avwct=areaaverage(WaterCatch,OutZones);        #Water content
report $4\lu_wct.tss=timeoutput(OutZones,lu_avwct);

lu_avevap=areaaverage(ActEvap,OutZones);                #evaporation
report $4\lu_evap.tss=timeoutput(OutZones,lu_avevap);
report $4\lu_potenevap.tss=timeoutput(OutZones,areaaverage(PotenEvap,OutZones));
report $4\lu_pottrans.tss=timeoutput(OutZones,areaaverage(PotTrans,OutZones));
report $4\lu_AtransMM.tss=timeoutput(OutZones,areaaverage(AtransMM,OutZones));
report $4\lu_Rs.tss=timeoutput(OutZones,areaaverage(Rs,OutZones));
report $4\lu_Ra.tss=timeoutput(OutZones,areaaverage(Ra,OutZones));
report $4\lu_VPD.tss=timeoutput(OutZones,areaaverage(VPD,OutZones));
report $4\lu_Delta.tss=timeoutput(OutZones,areaaverage(Delta,LandUse));
report $4\lu_AintMM.tss=timeoutput(OutZones,areaaverage(AtransMM,OutZones));
#report $4\lu_AsoilMM.tss=timeoutput(OutZones,areaaverage(AtransMM,OutZones));
report
         $4\lu_NetInterception.tss=timeoutput(OutZones,areaaverage(NetInterception,OutZone
         s));
report $4\lu_Interception.tss=timeoutput(OutZones,areaaverage(Interception,OutZones));
report $4\lu_Inwater.tss=timeoutput(OutZones,areaaverage(Inwater,OutZones));
report $4\lu_Rad.tss=timeoutput(OutZones,areaaverage(Radiation,OutZones));
report $4\lu_Windspeed.tss=timeoutput(OutZones,areaaverage(WindSpeed,OutZones));
report $4\lu_Temp.tss=timeoutput(OutZones,areaaverage(Temperature,OutZones));
report $4\AtransMM.tss=timeoutput(TopoId,areaaverage(AtransMM,TopoId));
report $4\AintMM.tss=timeoutput(TopoId,areaaverage(AtransMM,TopoId));
#report $4\AsoilMM.tss=timeoutput(TopoId,areaaverage(AtransMM,TopoId));

report
         $4\StatExposurePoints.tss=timeoutput(RainAreas,areaaverage(StatExposurePoints,Rai
         nAreas));
report $4\StatExposure.tss=timeoutput(RainAreas,areaaverage(StatExposure,RainAreas));

# some temp reporting of variables
#report $2\C=CanopyStorage;
#report $2\TF=ThroughFall;
#report $2\P=Precipitation;
#report $2\PP=Rain_slope;
#report $2\PPP=Rain_diff;
#report $2\WC=WaterCatch;
#report $2\ET=ActEvap;
#report $2\ETR=RestPotEvap;
#report $2\RC=RadCor;
#report $2\SW=ShortWave;
#report $2\FD=FirstZoneDepth;
#report $2\UC=UStoreCapacity;
#report $2\UD=UStoreDepth;
#report $2\PR=Radiation;

#########################
# Report totals in maps #
#########################
report($5) $2\PathInfiltExceeded.map=PathInfiltExceeded;
report($5) $2\SoilInfiltExceeded.map=SoilInfiltExceeded;
```

```
report($5) $2\CumReinfilt.map=CumReinfilt;
report($5) $2\CumInfiltExcess.map=CumInfiltExcess;
report($5) $2\CumExfiltWater.map=CumExfiltWater;
report($5) $2\CumPotPrec.map=CumPotPrec;
report($5) $2\CumPrec.map=CumPrec;
report($5) $2\CumPrecPol.map=CumPrecPol;
report($5) $2\CumPrecWindCor.map=CumPrecWindCor;
report($5) $2\CumEvap.map=CumEvap;
report($5) $2\CumPotenEvap.map=CumPotenEvap;
report($5) $2\CumInt.map=CumInt;
report($5) $2\CumWaterCatch.map=CumWaterCatch;
report($5) $2\CumLeakage.map=CumLeakage;
report($5) $2\Budget.map=CumPrec+CumWaterCatch-CumInt-CumEvap;
# report averages in maps
report($5) $2\AvRad.map=CumRad/$5;
report($5) $2\AvPrecAngle.map=CumPrecAngle/$5;
report($5) $2\AvHPPrecAngle.map=CumHPPrecAngle/$5;
report($5) $2\AvInfiltExcess.map=CumInfiltExcess/$5;
report($5) $2\AvSurfaceWater.map=CumSurfaceWater/$5;
report($5) $2\AvReinfilt.map=CumReinfilt/$5;

report($5) $2\AvCorrectedExposureWindSpdRatio.map = CumCorrectedExposureWindSpdRatio/$5;
report($5) $2\AvExposureWindSpdRatio.map = CumExposureWindSpdRatio/$5;
report($5) $2\AvWindSpeed.map = CumWindSpeed/$5;
report($5) $2\AvWindSpeedStation.map = CumWindSpeedStation/$5;

#report $2\AvailCap.map=AvailCap;
#report $2\zi.map=zi;



################################################
# report state variables at end and during a run #
################################################
# Canopy...
report($5) $2\CanopyStorage.map=CanopyStorage;

# Soil...
report($5) $2\FirstZoneDepth.map=FirstZoneDepth;
report($5) $2\UStoreDepth.map=UStoreDepth;
report($5) $2\SurfaceRunoff.map=SurfaceRunoff;
report($5) $2\SumCellWatBal.map=SumCellWatBal;
```

Scenario run results

Introduction

The section describes the results of applying the cqflow model. The model has been run for the entire Rio Chiquito catchment and for a cloud forest sub catchment in which the outlet of the catchment has been placed at 1400 m. The following scenarios model runs are described:

1. Now. This model is run with the present day land cover. Present day land cover has been derived from the 2001 Landsat image.
2. Removal of the cloud forest. In this run all the forest above 1400m has been converted into pasture.
3. Forest. In this model run most of the catchment is covered with forest. This forest cover was taken from the 1975 Landsat image.
4. Pasture. In this model run most of the forest has been removed.
5. Cloud forest catchment, present day land use
6. Cloud forest catchment, 1975 Land use
7. Cloud forest catchment, complete conversion to pasture

In all the runs all the model parameters have remained identical. The only thing that has been changed is the land-use map. The cloud forest runs are performed using a subcatchment of the Rio Chiquito.

Limitations

The following (and other) limitations apply to these results:

- Very little calibration of the soil subsystem has been performed.
- The parameters Rs and Ra in the evaporation function have been derived from cloud forest measurements and may not be applicable to the lower regions of the catchment.
- A leakage term has been introduced that remove a fraction of the water that transfers from the unsaturated to the saturated component.

NB All calculations were performed for the period 1 July 2003 to 1 July 2004 using hourly time steps. some of the time axes in the graphs do not show the correct date.

Water budget of the Rio chiquito catchment

This section describes the water budget of the Rio Chiquito catchment using a combination of measured and modelled data. The following component of the water budget have been determined:

1. **Incoming precipitation (P).** This is the precipitation derived from the gauge data and converted into an areal mean for the catchment using the methods described in the previous chapters. This includes correction for wind-speed and -direction, precipitation angle and aspect of the terrain.

2. **Horizontal precipitation (HP)**. This is defined as the precipitation coming into the catchment at an angle greater than 85 degrees. Corrections for catching efficiency and wind speed are applied (see previous chapters).
3. **Transpiration (Et).** Transpiration from the vegetation as determined using the Penman-Monteith equation.
4. **Interception (Ei)**. Rainfall interception (evaporation from the wet canopy) as determined by a simplified Rutter-type interception model (see description of the model).
5. **Discharge (Q)**. Discharge as ***measured*** at the catchment outlet.
6. **Delta storage (S).** The change in storage in the catchment in the period used for the water budget calculations as determined from he present day model run.
7. I**nferred leakage (L).** Leakage as determined from the difference between the input terms and the output term of the water budget.

The water budget for the catchment can be written as:

P + HP = Q + Et + Ei + S + L

As no measurements of leakage are available this amount has been determined as a rest term. The following table shows the water budget for the period July 2003 to July 2004 (8760 hrs):

Parameter	Amount [mm]
P	4680
HP	589
Et	467
Ei	488
Q	2909
S	151
L	1255

Table 1: Water budget for the Rio Chiquito catchment (July 2003 to July 2004)

Note that modelled discharge and leakage determined using modelled discharge were found to be different at 3501 and 663 mm respectively. Increasing the leakage component in the model may narrow this gap but a short analysis revealed that this did not significantly increase model performance and no further calibration attempts have therefore been made.

Measured discharge is not available for the subcatchment runs. Therefore, the water budget for the subcatchment has been based on model results only. Because the whole basin water budget indicates that the model under-estimated the leakage term by approximately 100% the budget has also been calculated by doubling the amount of leakage. This is shown by the numbers in brackets.

Parameter	Amount [mm]
P	5369
HP	735
Et	378
Ei	551
Q	4339 (3636)
S	133
L	703 (1406)

Table 2: Water budget for the Cloud Forest subcatchment (July 2003 to July 2004)

PRESENT SITUATION

The figure below shows the land-use map for the present day scenario. The map is based on a 2001 Landsat image.

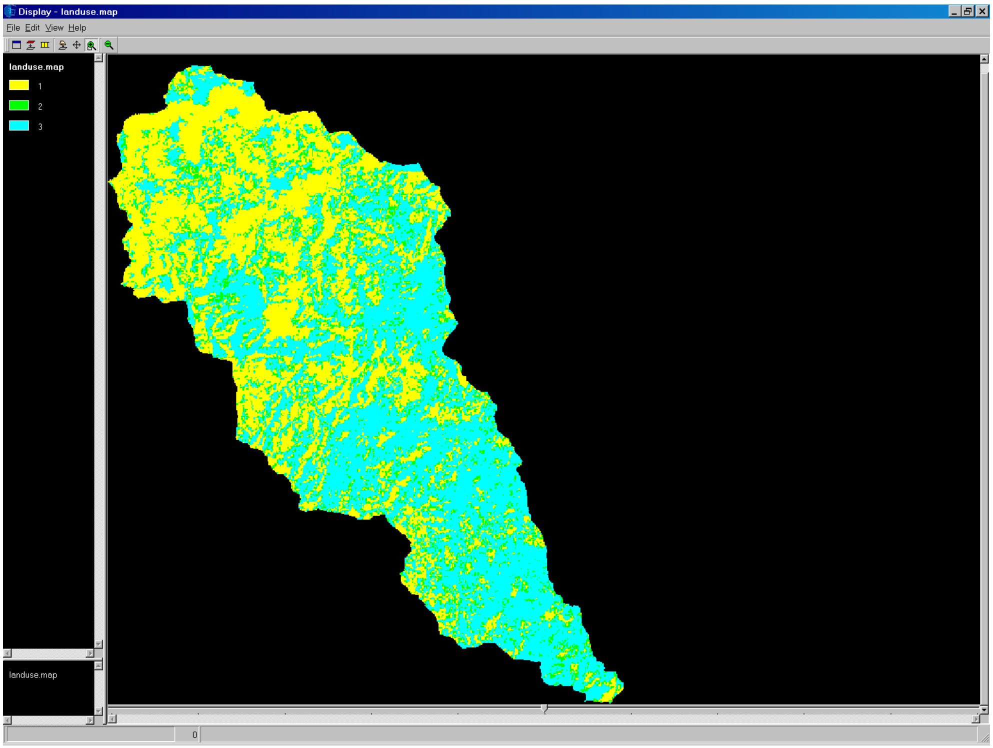


Illustration 8 Landuse map for the present day situation. Yellow = pasture, Green = Secondary Forest, Cyan = Primary Forest

The modelled water budget for this run is shown in the table below:

```
Model eff:              0.658252
Precipitation:          4680
Fog interception:       589
Throughfall:            4758
Interception:           487
```

```
Evaporation:              467
Runoff:                   3501
Deep leakage:             662
Change in storage:        150
```

Selected results from the now-run

The runs with the present day conditions has been used to establish the performance of the model. Initial runs have been made using a low resolution grid (150x150m) to facilitate calibration. The final runs (discussed in this report) have been made using a 25x25 meter grid. Running the model for one year takes about 36 hours on a 3.9Ghz pc using the high resolution grid and about 1 hour using the low resolution grid.

Modelled discharges compares relatively well to observed discharge as can been seen in Illustration 10. Model efficiency is about 0.66 (Nash and Sutcliff). Given the limited amount of calibration of the soil subsystem this is a good fit.

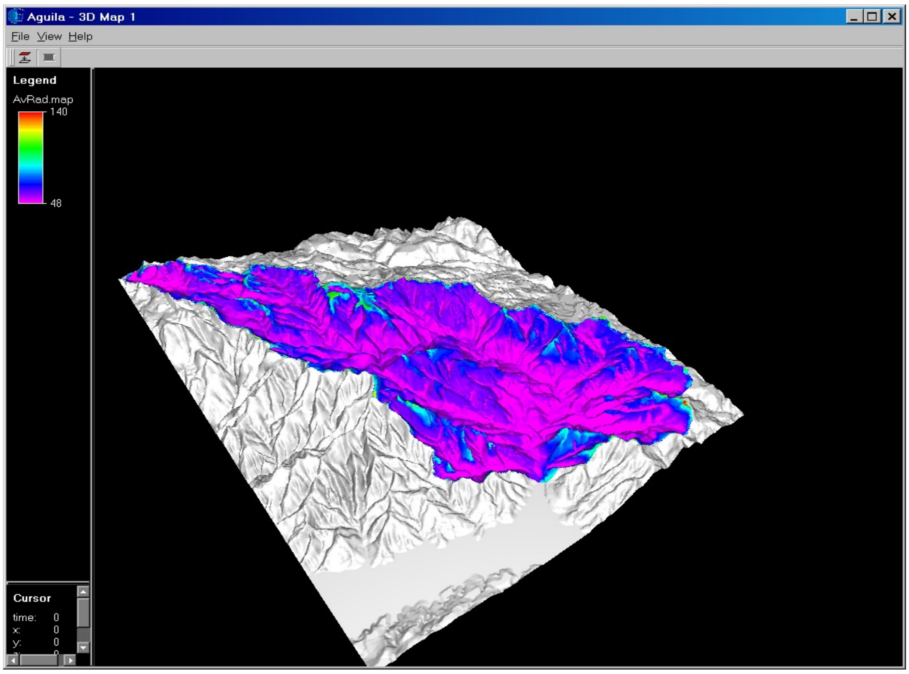


Illustration 9 Average radiation over the catchment in W/m^2

Average content of the saturated and unsaturated storage component in the model is shown in Illustration 11.

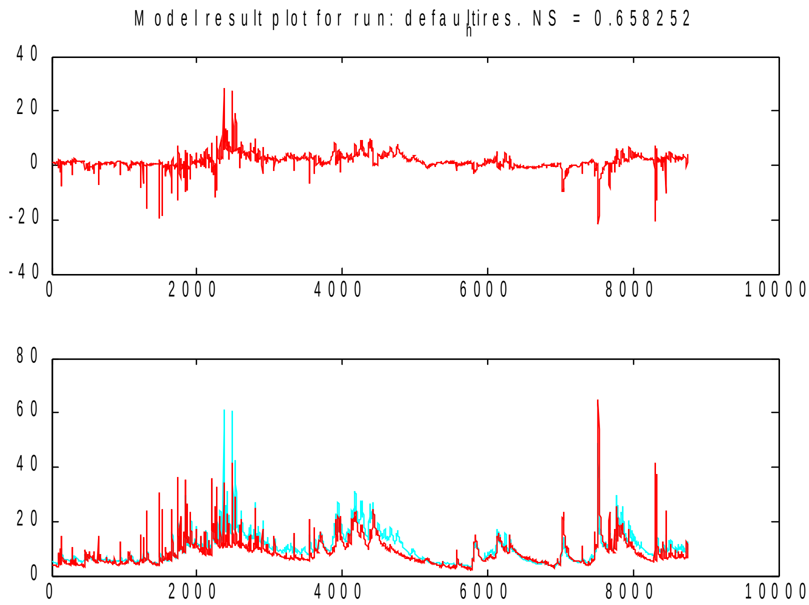


Illustration 10Measured (red, bottom part of the figure) and modelled discharge (both in mm/hr) for the present day situation. The top part shows the difference between measured and modelled discharge. Nash and Sutcliff model efficiency is about 0.66

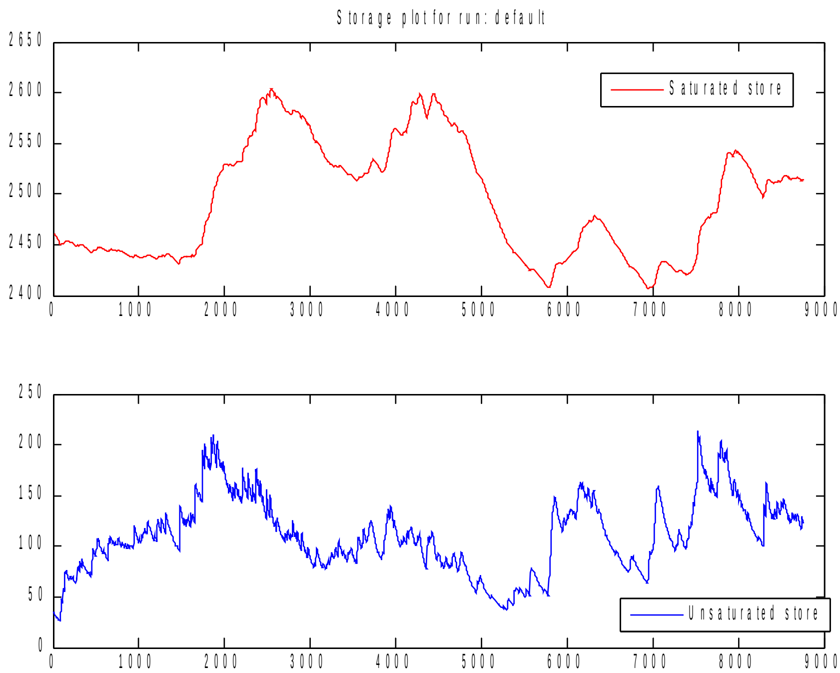


Illustration 11: Content of the saturated and unsaturated stores in the model. Y-axis unit is mm, x-axis is hours since 1 July 2004

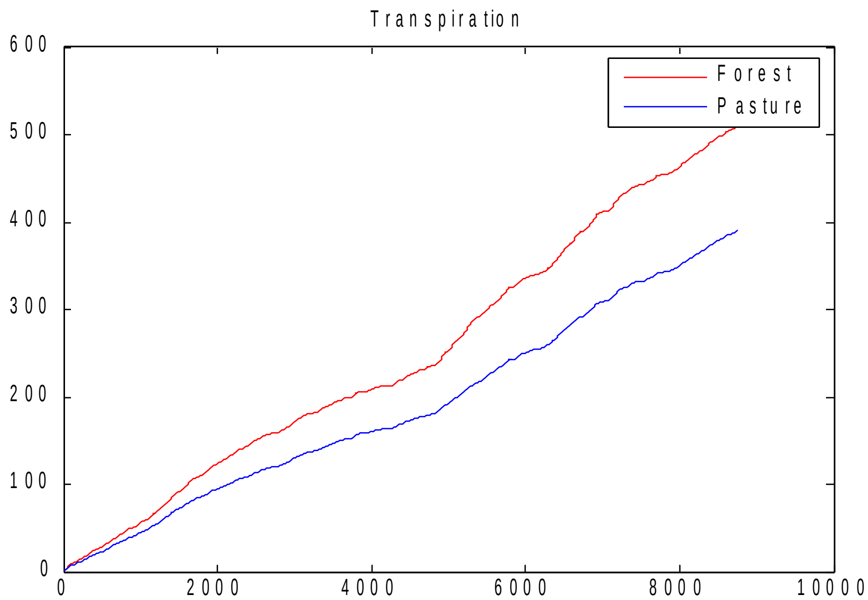


Illustration 12 Cumulative area average Evaporation (dry canopy, mm) for forest and pasture land cover. X-axis shows hours since 1 July 2003.

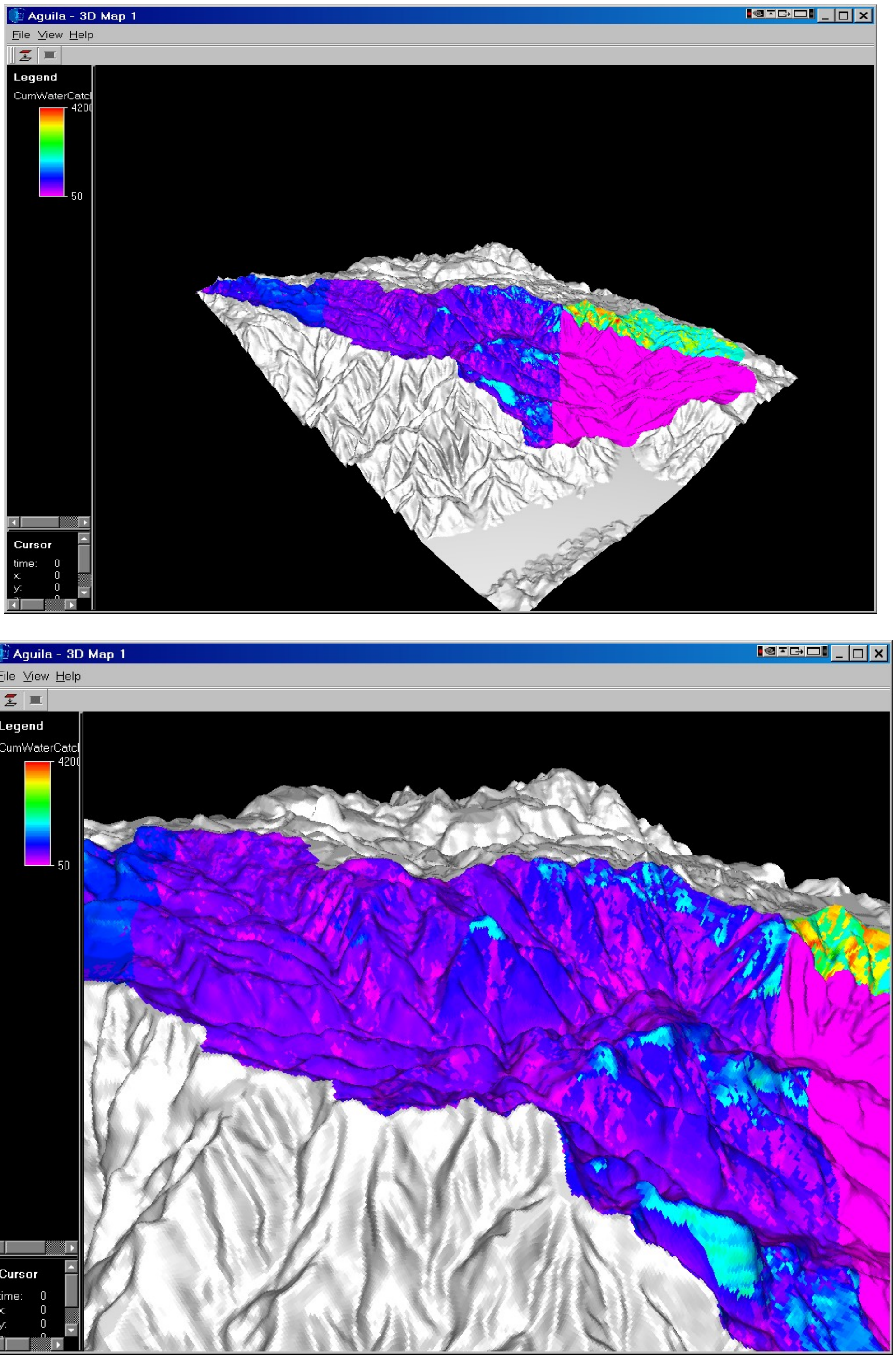


Illustration 13 Total horizontal precipitation in mm (Angle > 85 degrees). The top figure shows the whole catchment, the bottom figure shows a detail in the cloud forest part

The model also calculates the amount of infiltration excess flow. Note that not all this water necessarily ends up in the main channel. If unsaturated nodes are available downstream re-infiltration occurs (Illustration 14).

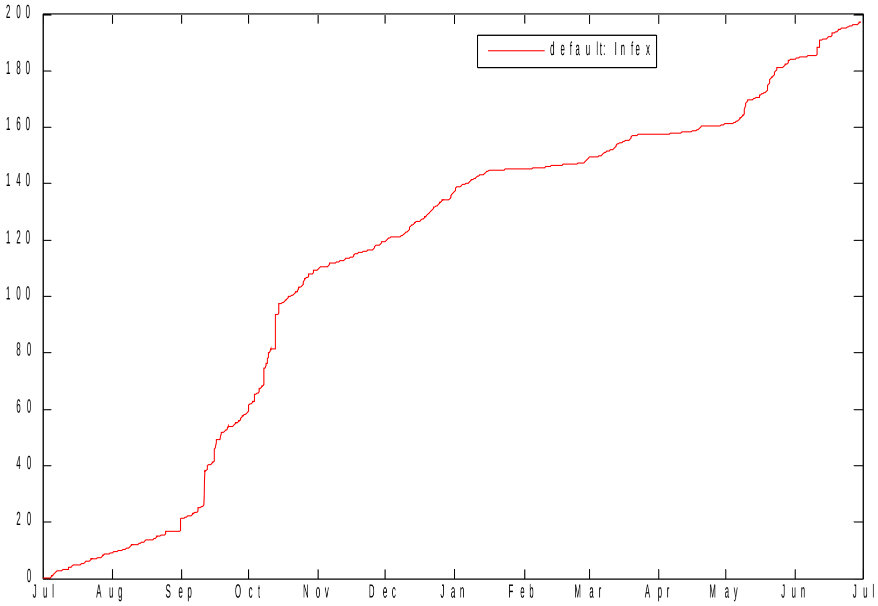


Illustration 14: Infitration excess in mm as determined by the model

REMOVAL OF THE CLOUD FOREST ONLY

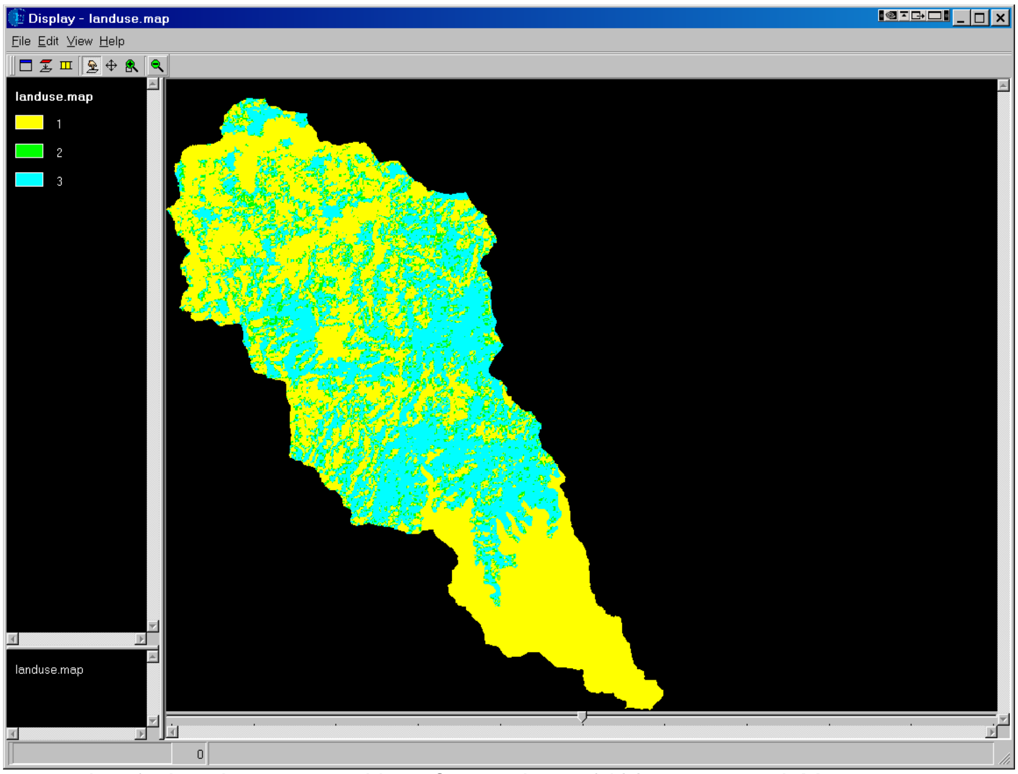


Illustration 15 Land-use map with all forest above 1400 m removed. Yellow = pasture, Green = Secondary Forest, Cyan = Primary Forest

In this scenario all the forest cover at altitudes higher than 1400 m has been converted to pasture (Illustration 15). This resulted in the following water budget:

Precipitation:	4680
Fog interception:	545
Throughfall:	4760
Interception:	441
Evaporation:	453
Runoff:	3515
Deep leakage:	664
Change in storage:	151

The effect of cloud forest removal is relatively small. One reason for this is that some of the most exposed and windy sites of the catchment (that receive a lot of HP) are below 1400m.

Illustration 16 Shows the difference in discharge between the now situation and the scenario in which all forest above 1400m has been converted to pasture. A significant difference can bee seen in the amount of infiltration excess water. Although most of this water re-infiltrates downslope it does give an indication of the occurrence of erosive overland flow in the catchment (Illustration 17).

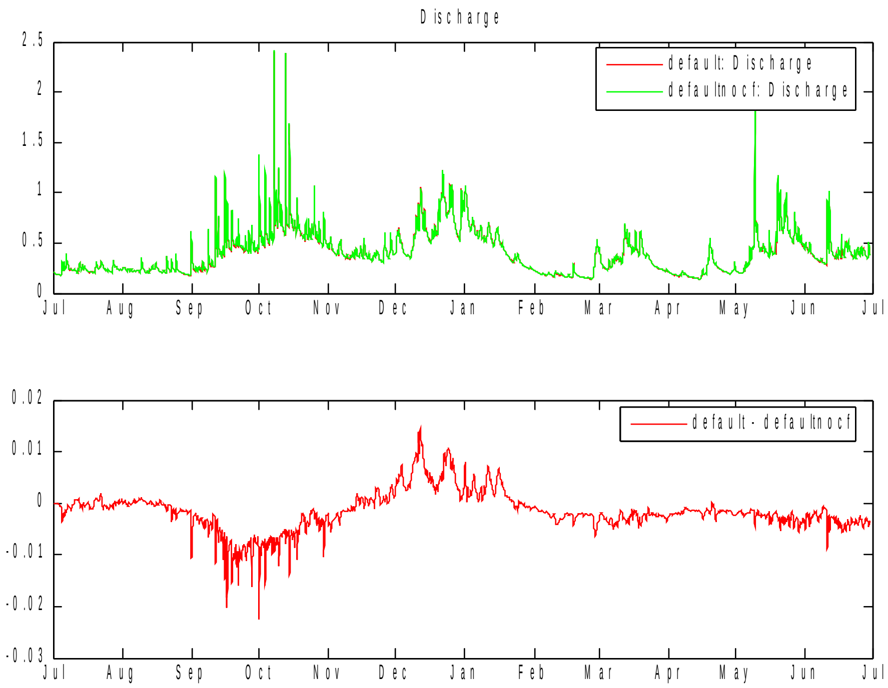


Illustration 16 Discharge in mm/hr for the now situation and the scenario in which all forest above 1400 m has been removed. The bottom part of the figure shows the difference in discharge

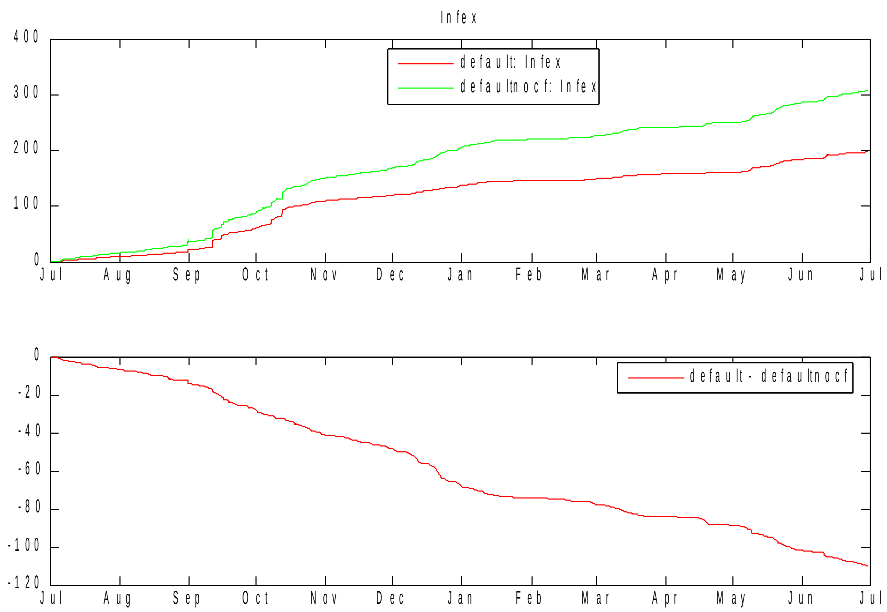


Illustration 17: Difference in amount of infiltration excess (mm) water between the current situation and when removing all CF above 1400 m

FOREST

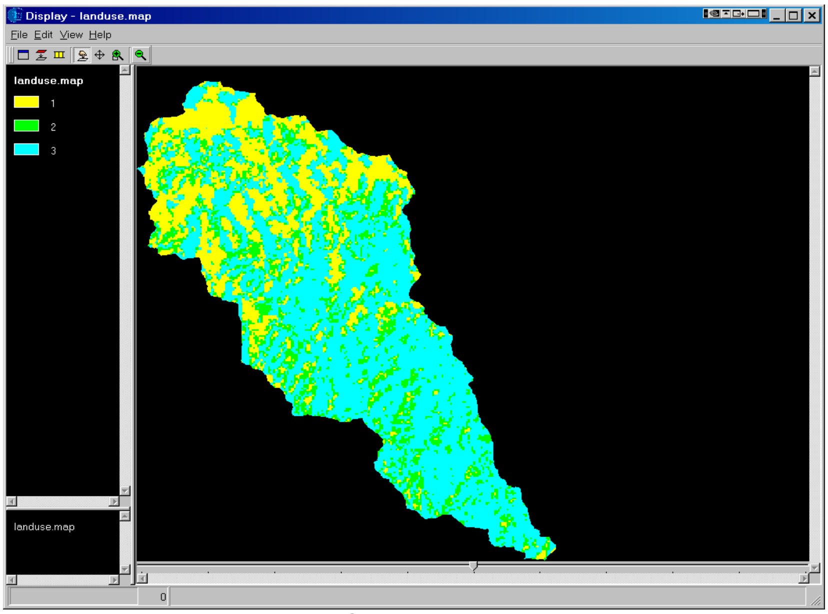


Illustration 18 Land-use map for the 1975 situation. Yellow = pasture, Green = Secondary Forest, Cyan = Primary Forest

In this scenario the land use for 1975 (mostly forested) has been used in modelling. This resulted in the following water budget:

Precipitation:	4680
Fog interception:	633
Throughfall:	4747
Interception:	541
Evaporation:	491
Runoff:	3473
Deep leakage:	657
Change in storage:	149

The difference in discharge between the now and 1975 situation is shown in Illustration 19.

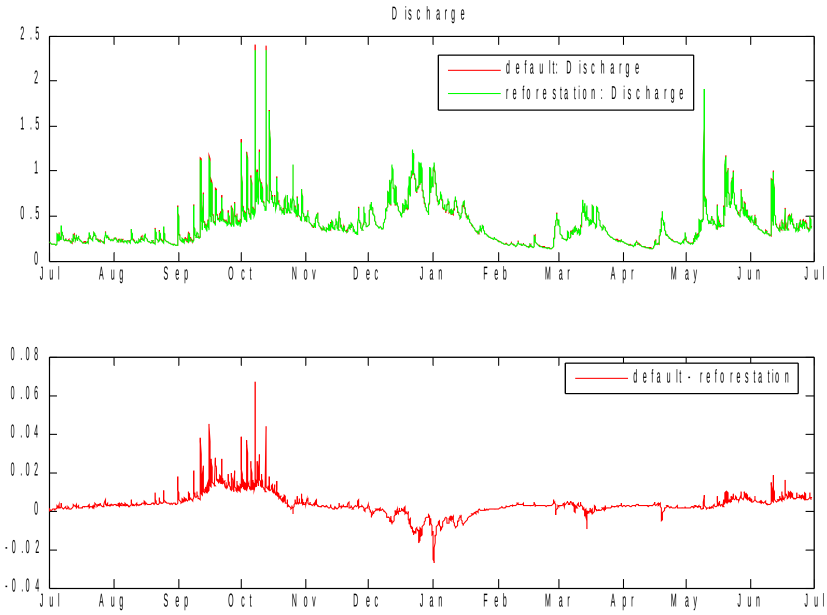


Illustration 19 Discharge (mm/hr) for the now situation and the scenario in which the 1975 land cover has been restored. The bottom part of the figure shows the difference in discharge.

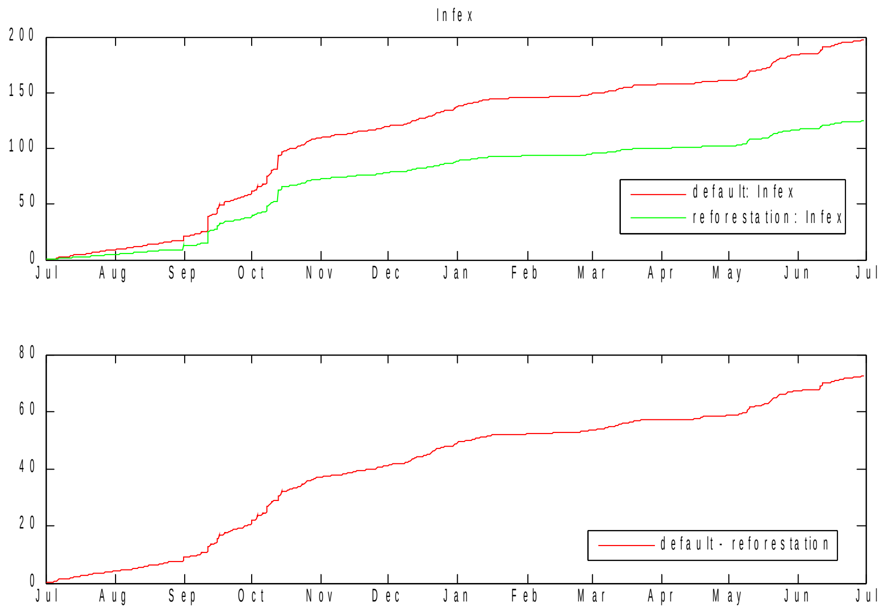


Illustration 20: Difference in infiltration excess water (mm) between the now and pasture scenario's

PASTURE

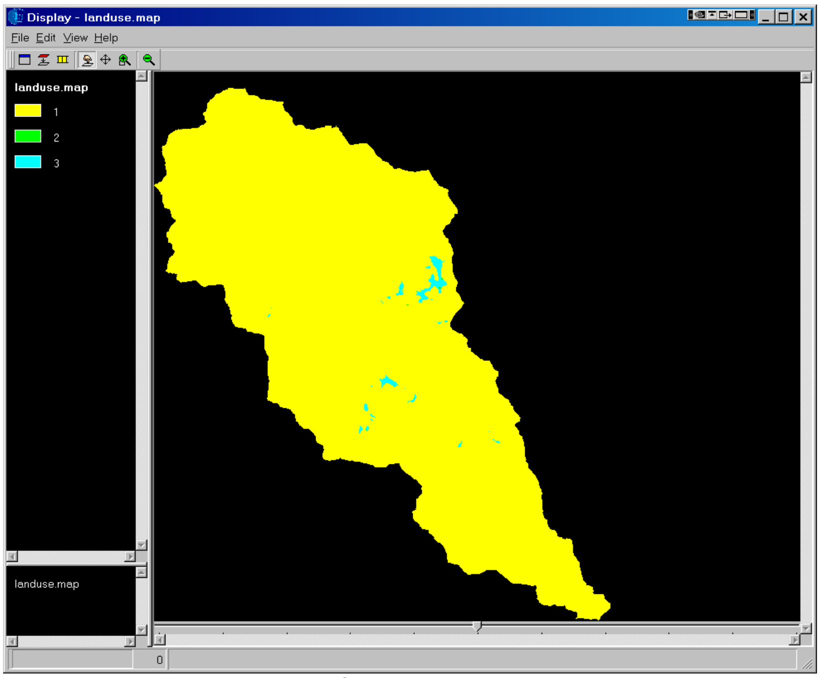


Illustration 21Land-use map for an almost entirely cleared catchment. Yellow = pasture, Green = Secondary Forest, Cyan = Primary Forest

In this scenario almost all forest has been converted to pasture resulting in the following catchment water budget:

```
Precipitation:          4680
Fog interception:       430
Throughfall:            4814
Interception:           272
Evaporation:            373
Runoff:                 3625
Deep leakage:           682
Change in storage:      156
```

The effect of converting all the forest to pasture on discharge is shown in Illustration 22 while Illustration 23 shows the effect on infiltration excess water.

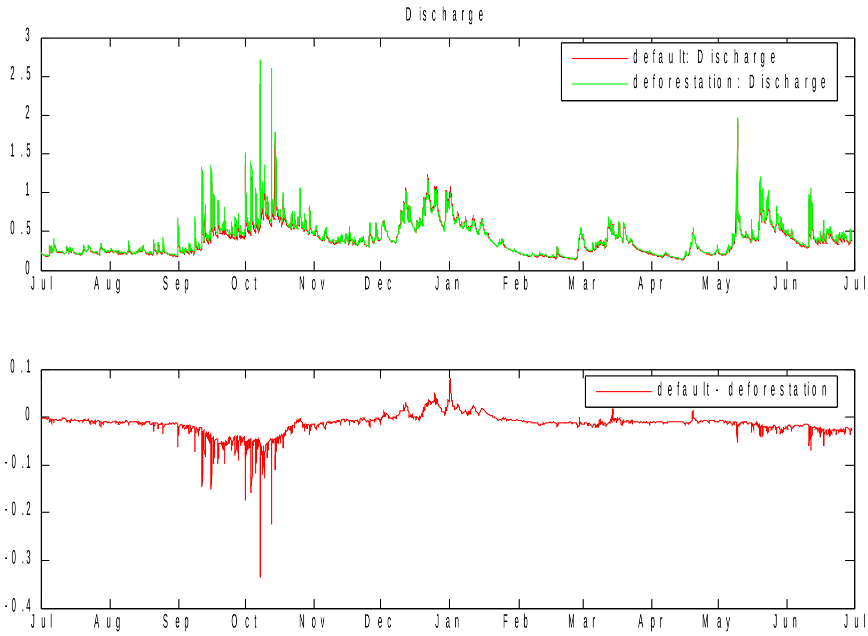


Illustration 22 Discharge (mm/hr) for the now situation and the scenario in which most of the forest has been removed. The bottom part of the figure shows the difference in discharge

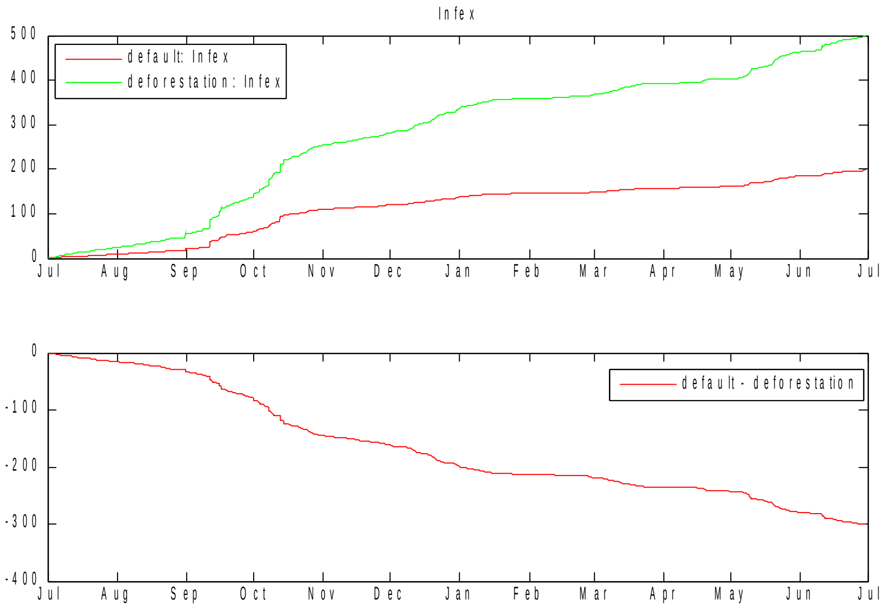


Illustration 23: Difference in infiltration excess water (mm) between the now and deforested (pasture) scenario's

Subcatchment runs

Introduction

For the following runs a new catchment was defined that is located mostly in the zone above 1400 m. The (virtual) gauge at the outlet is located at 1321 m. A map of the subcatchment is shown in Illustration 24. The catchment is relatively sheltered for the most common wind directions.

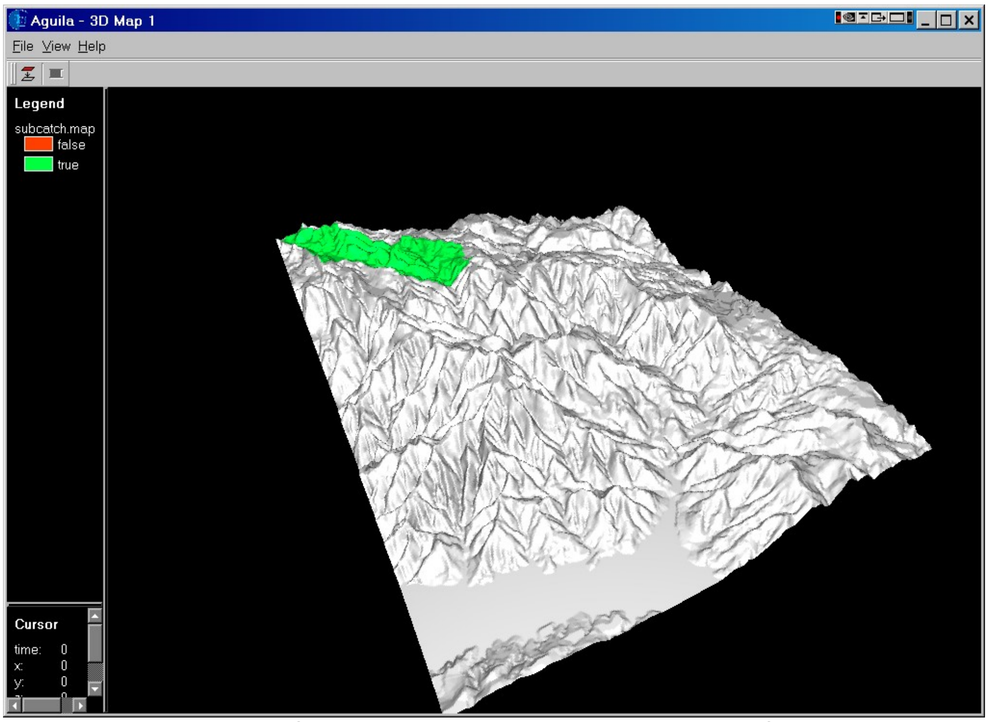


Illustration 24 The cloud forest subcatchment that was extracted for the subcatchment runs

Now situation

The following table show the water budget results (in mm) for the cloud forest subcatchment using the present landuse.

Precipitation:	5369
Fog interception:	735
Throughfall:	5526
Interception:	550
Evaporation:	378
Runoff:	4339
Deep leakage:	702
Change in storage:	133

Compared to the whole catchment the contribution of HP is significantly higher. In this case net interception is negative, i.e. the forest catches more HP than the total amount of wet canopy evaporation.

Compared to the total basin streamflow expressed in mm/hr is higher most of the time (Illustration 25) although some of the high peaks in October 2003 are lower in the CF subcatchment. This is because the rainfall peaks at that time are localized to some of the lower rain gauges.

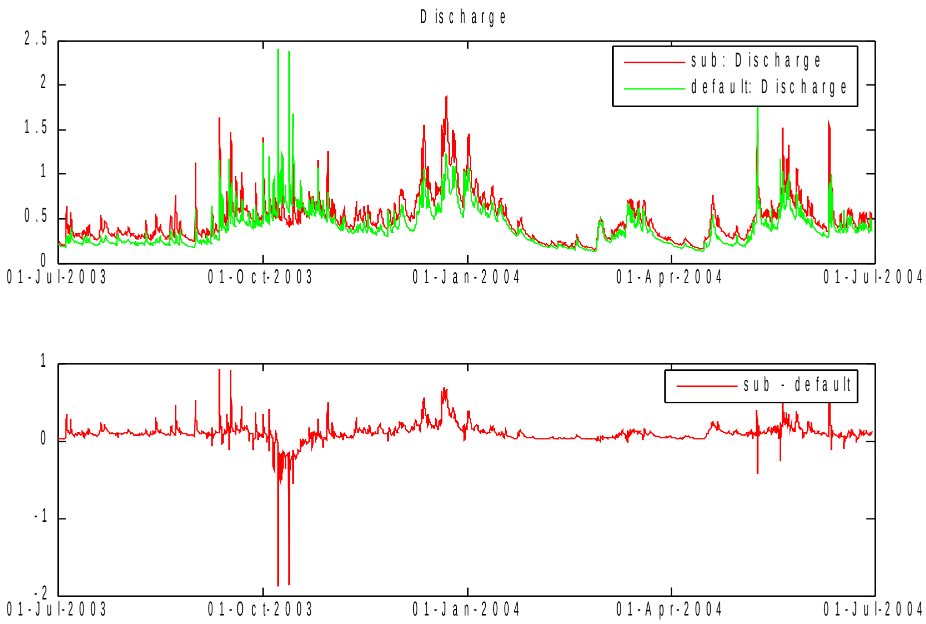


Illustration 25: Discharge (mm/hr) of the cloud forest sub catchment compared to the whole Rio Chiquito catchment

All forest removed (Pasture)

The following table shows the results (in mm) for the cloud forest subcatchment when all forest has been converted to pasture.

Precipitation:	5369
Fog interception:	473
Throughfall:	5532
Interception:	284
Evaporation:	298
Runoff:	4414
Deep leakage:	710
Change in storage:	135

The resulting discharge is shown in Figure 26 together with the discharge obtained from the run with the current landuse.

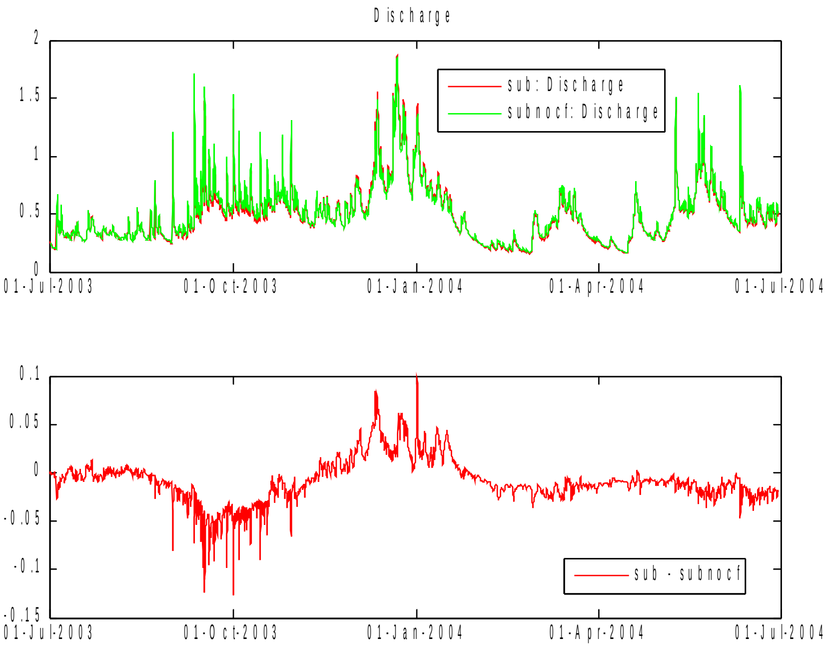


Illustration 26 Discharge (mm/hr) for the default run (current situation) and the run in which all forest has been converted to pasture.

A cumulative version of the graphs is shown in Figure 27.

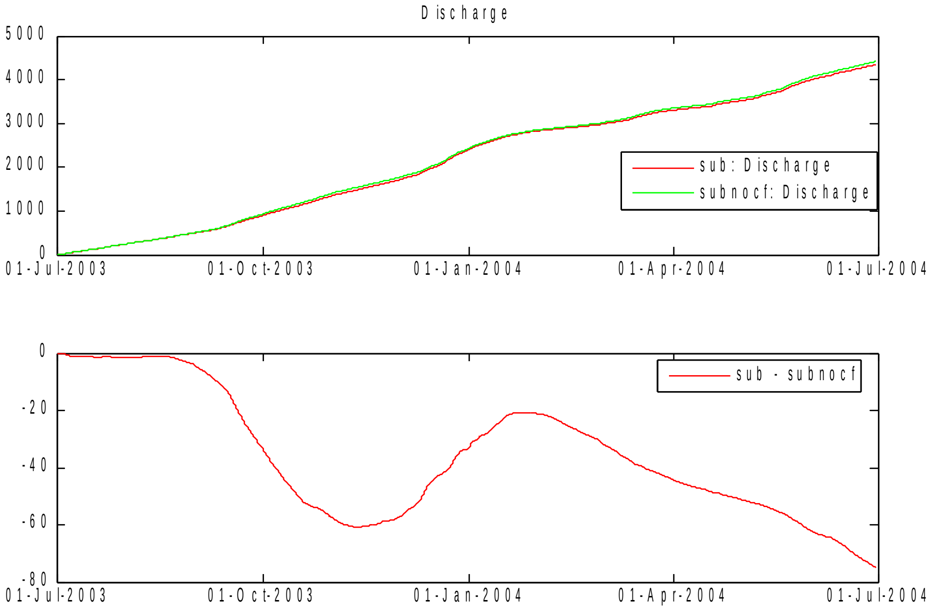


Illustration 27 Cumulative discharge (mm) for the run with the current situation and the run when the cloud forest has been converted to pasture.

Although the effect of the increasing infiltration excess water is hardly reflected in the discharge at the subcatchment outlet the decreased infiltration capacity of the pasture makes a relatively large difference at the local scale (Illustration 28). As stated before (see also the description of the model code in the previous chapter) this is water that cannot infiltrate on the most compacted parts of the soil (e.g. Cattle trails). However, most of this water re-infiltrates; either in the same cell on the non compacted part or in a downstream cell. Although very little of this water ever reaches the main channels, it does indicate that there is a larger potential for erosion when the forest is replaced by pasture.

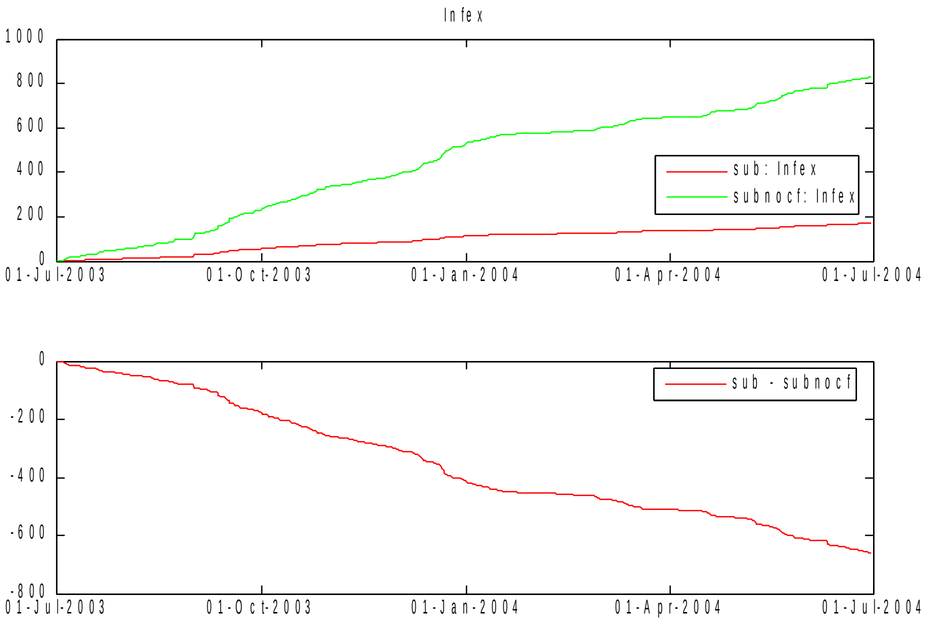


Illustration 28: Difference in infiltration excess water (mm) between the current situation and the pasture situation.

1975 Forest cover

The following table shows the results (in mm) for the cloud forest subcatchment when the land-use is nearly completely forested.

Precipitation:	5369
Fog interception:	778
Throughfall:	5521
Interception:	599
Evaporation:	393
Runoff:	4321
Deep leakage:	701
Change in storage:	133

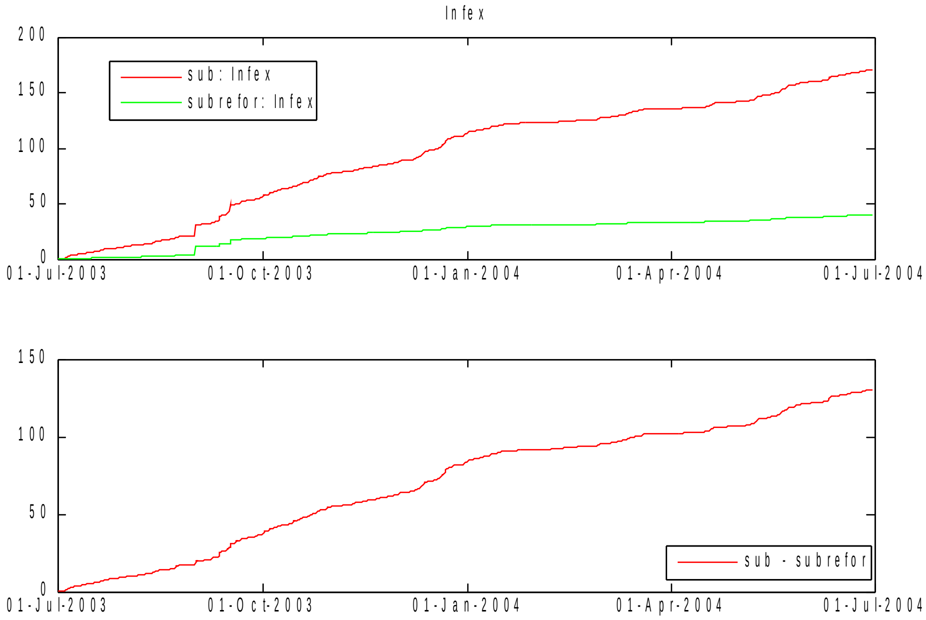


Illustration 29: Difference in infiltration (mm) excess between current and 1975 (forested) situation

Summary of results

Water budget results

The following tables summarize the water budget results obtained with the different scenario runs. The % columns give the percentage compared to the current situation.

Whole catchment results (pasture CE = 15%, forest CE = 25%)

Scenario	P	HP		Et		Ei		Q		Leakage		Storage
	mm	mm	%	mm	%	mm	%	mm	%	mm	%	
Current	4680	589		467		488		3501		663		150.73
1975 cover	4680	633	107	491	105	542	111	3474	99	658	99	149.25
Pasture	4680	430	73	373	80	273	56	3625	104	683	103	156.86
No cloudforest	4680	545	92	453	97	442	91	3515	100	664	100	151.15

In the subcatchment runs the model was run for the cloud forest subcatchment only. The table below shows the results:

Cloud forest subcatchment results (pasture CE = 15%, forest CE = 25%)

Scenario	P	HP		Et		Ei		Q		Leakage		Storage
	mm	mm	%	mm	%	mm	%	mm	%	mm	%	
Current	5369	735		378		551		4339		703		133.26
Forested (1975)	5369	779	106	393	104	599	109	4321	100	701	100	132.73
Pasture	5369	473	64	298	79	285	52	4414	102	711	101	135.10

At a first glance these results indicate that there is hardly any change in the water budgets of the subcatchment and the whole catchment after (cloud) forest conversion to pasture. The relatively small changes in discharge -- 4321 mm for the forested scenario vs. 4413 mm for the pasture scenario in the case of the cloud forested subcatchment-- are caused by the fact that the decreased horizontal precipitation HP input associated with pasture is balanced out by the concomitant decrease in evaporation losses. For the whole Rio Chiquito catchment the difference in evaporation losses between forest and pasture is smaller but so is the difference in HP, leading to a similar result.

Effects on catchment wetness

Within the catchment the decreased evaporation losses when converting forest to pasture result in a slightly wetter soil during most of the year. Illustration 30 shows the average water depth of the saturated store of the forested and pasture runs for the whole catchment. The pasture situation results in a slightly wetter soil. This difference can also be seen in Illustration 31 and Illustration 32 that show the difference in saturated catchment percentage for the forested and pasture scenario's. Illustration 31 and Illustration 32 indicate that the cloud forest subcatchment has a larger saturated area compared to the whole catchment.

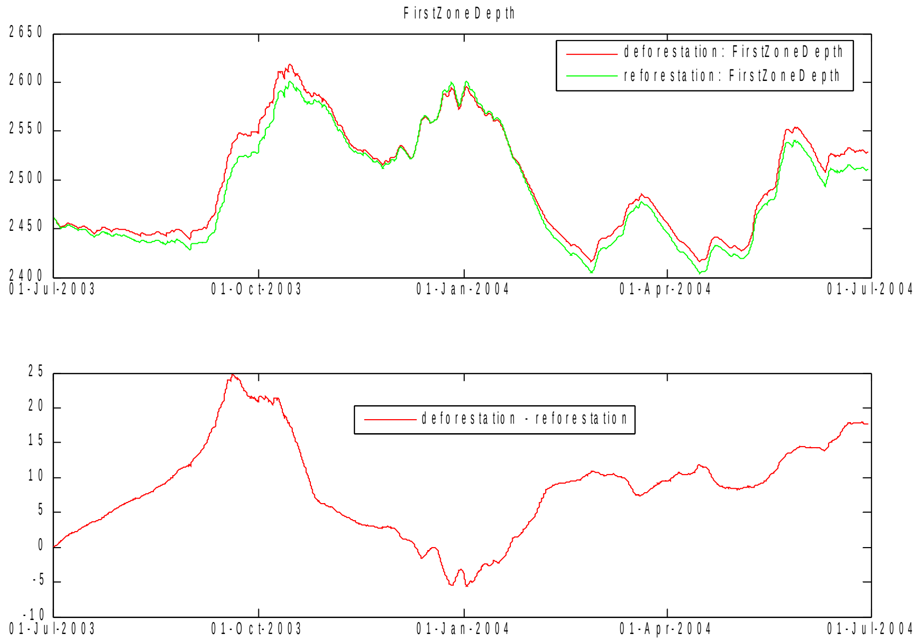


Illustration 30: Average water content in mm of the saturated store for the whole catchment

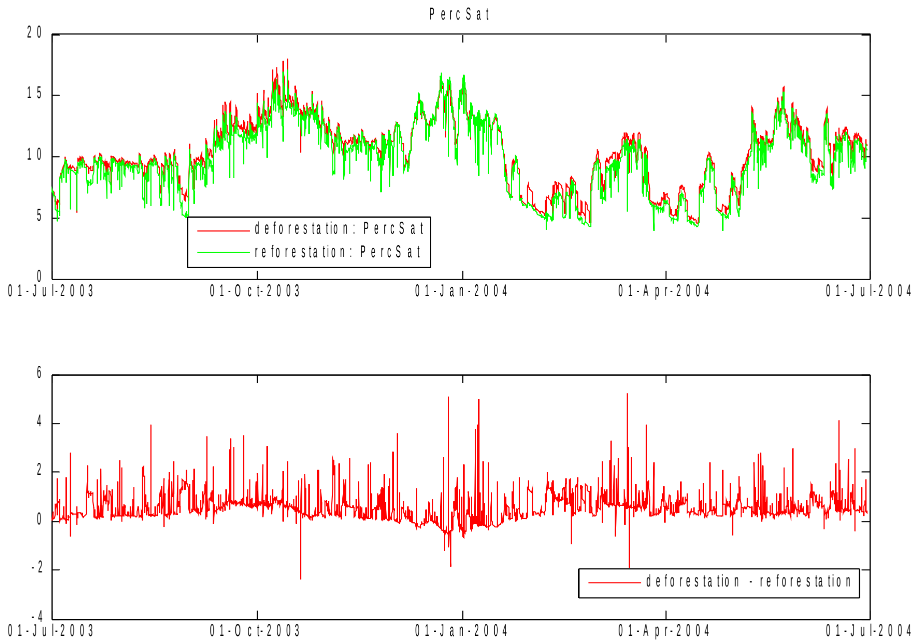


Illustration 31: Percentage of the whole catchment that is saturated for the forested and pasture scenario. The lower panel shows the difference between the two scenario's.

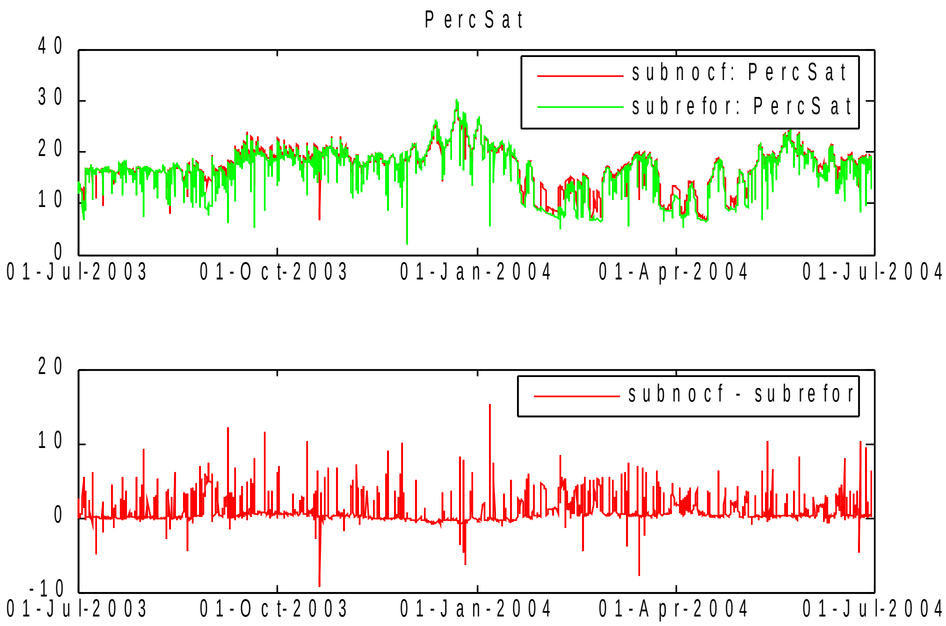


Illustration 32: Saturated percentage for the cloud forest subbasin comparing the pasture and forested scenarios The lower panel shows the difference between the two scenario's.

EFFECTS ON INFILTRATION EXCESS WATER

Within the model a sub-cell parameterization exists that determines the fraction of compacted soil (trails) in each cell. For both the compacted part and the undisturbed part a conductivity value may be parametrized. These parameter values are shown below.

parameter	***Pasture [mm/hr]***	***Forest [mm/hr]***
Compacted area fraction	0.2	0
Ksat compacted	20	45
Ksat non-compacted	45	45

If the infiltration capacity of the compacted area is exceeded by rainfall intensity, this is recorded as infiltration excess water (overland flow). Within the same time step this water may re-infiltrate in the non-compacted area within the same cell if this has enough capacity. In subsequent time steps any remaining water is routed downslope using the kinematic wave routine where it may re-infiltrate if capacities allow.

Therefore, the amount of infiltration excess that is modelled is not necessarily directly related to quick runoff volumes in the main channel. This effect can be illustrated by comparing the differences in quick runoff with the differences in infiltration excess between the respective scenarios.

Quickflow and baseflow were separated using a straight line method. The slope of the line was fixed at 0.005 mm/h. The table below lists the quick- and baseflow percentages obtained using this method combined with the (local) infiltration excess values. Clearly the increase in infiltration excess water when converting to pasture (see also Illustration 33) is not reflected in a similar increase in quickflow percentage. Part of the water re-infiltrates downstream (in another cell) but most can be infiltrated in the undisturbed part of the same cell (not shown in the table).

Scenario	***Quickflow %***	***Baseflow %***	***Total infiltration excess [mm]***	***Downstream re-infiltration***
Current	6.9	93.1	197	91
Pasture	7.1	92.9	498	101
Forested	6.9	93.1	125	89
Sub-Current	8.0	92.0	170	150
Sub-Pasture	8.1	91.9	830	158
Sub-Forested	8.0	92.0	40	148

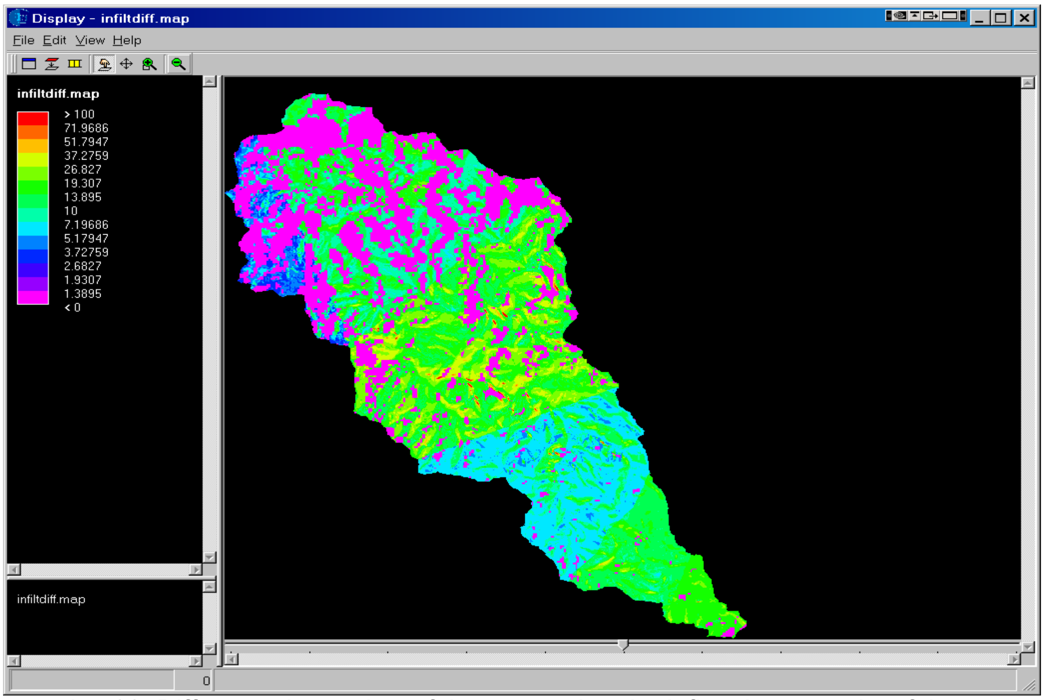


Illustration 33: Difference in number of time steps that the infiltration capacity of the (compacted) soil has been exceeded between the forested (1975) scenario and the pasture scenario. A larger value indicates the pasture scenario has a large number of time steps in which the infiltration capacity of the compacted part of the soil has been exceeded.

Illustration 34 shows quickflow traces for the subcatchment runs for the pasture and forest situation.

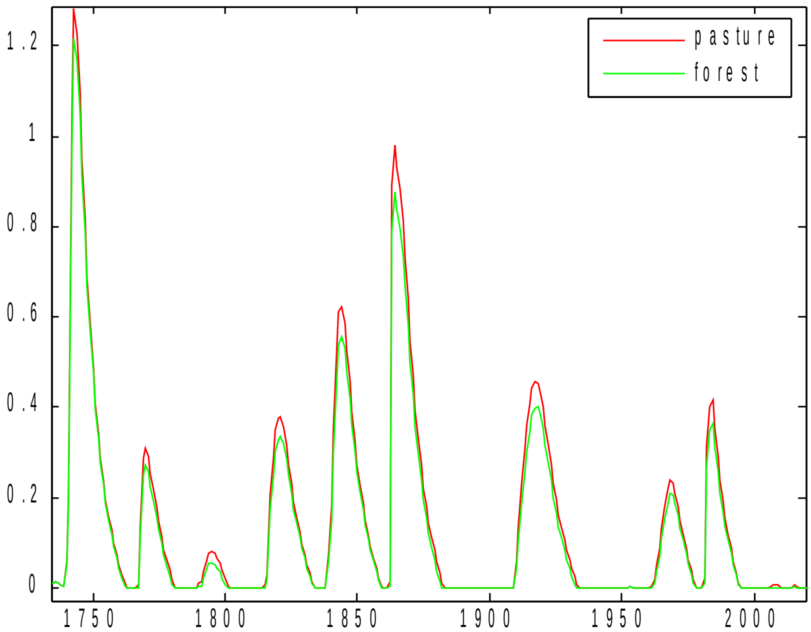


Illustration 34: Quickflow amounts (mm/hr) for the subbasin runs for the forested (green) and pasture scenario. The x-axis shows hours since 1 July 2003

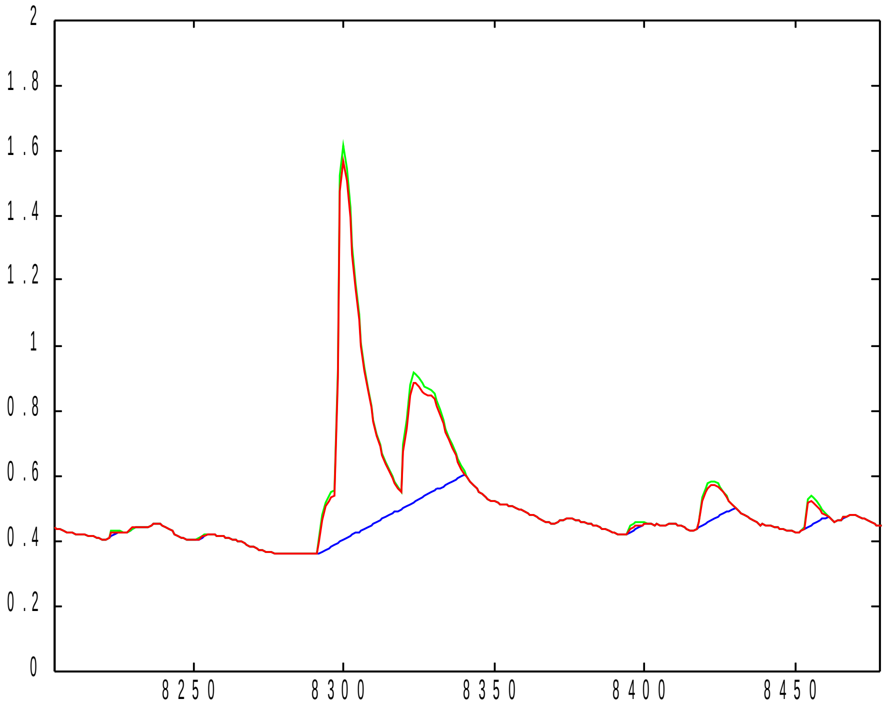


Illustration 35: Short period showing quickflow in mm/hr (red is forested, green is pasture) and baseflow (blue) using the straight line separation method for the subbasin runs. The x-axis shows hours since 1 July 2003.

MODEL PARAMETERS

This section lists the model parameter values that have been used in the respective runs. Please note that this list may be out of date. The actual model code in combination with the *.tbl files in the intbl directory always contain the most up to date version of any parameter.

In these files the number in the first column denotes the land-use type, the second column holds the actual value.

1: Pasture

2: Secondary forest

3: Primary forest

```
Albedo.tbl Albedo of the vegetation
1        0.183
2        0.135
3        0.135

Beta.tbl  Beta in the Topog SBM soil function
1        0.45
2        0.45
3        0.45

CanopyGapFraction.tbl Fraction of water that directly reaches the soil
1        0.05
2        0.05
3        0.05

CatchEff.tbl Catching efficiency for horizontal precipitation
1        0.15
2        0.25
3        0.25

FirstZoneCapacity.tbl Maximum water holding capacity of the saturated store (mm)
1        2900
2        2900
3        2900

FirstZoneKsatVer.tbl  Vertical saturated conductivity (mm/hr)
1        53
2        53
3        53

InfiltCapPath.tbl Maximum infiltration capacity of the compacted soil (mm/hr)
1        20
2        45
3        45

InfiltCapSoil.tbl  Maximum infiltration capacity of the undosturbed soil (mm/hr)
1        45
2        45
3        45

KsatHor.tbl  Horizontal saturated conductivity (mm/hr)
1        83
2        83
3        83
```

```
LeafAreaIndex.tbl Leaf area index (presently only used to estimate G)
1        2
2        5
3        6

M.tbl   Scaling parameter for the TOPOG_SBM functions
1        160
2        160
3        160

MaxCanopyStorage.tbl  Maximum amount of water that the canopy can hold (mm)
1        0.4
2        1.4
3        1.4

N.tbl  Manning N for the kinematic wave routine
1        0.6
2        0.6
3        0.6

PathFrac.tbl    Fraction of the cell that contains compacted soil
1        0.2
2        0
3        0

RootingDepth.tbl   Maximum depth at which root may extract water (mm)
1        300
2        600
3        600

VegetationHeigth.tbl  Height of the vegetation in meter
1        0.3
2        20
3        20

WindSpeedHeigth.tbl  Height at which has been measured
1        2.35
2        24
3        24

thetaR.tbl    Residual water content  m^3/m^3
1        0.1
2        0.1
3        0.1

thetaS.tbl   Saturated water content  m^3/m^3
1        0.6
2        0.6
3        0.6
```

DISCLAIMER

'This publication is an output from a research project funded partly by the United Kingdom Department for International Development (DFID) for the benefit of developing countries. The views expressed are not necessarily those of DFID [R7991 Forestry Research Programme]'.