# Computer-Based Calibration Methods for Parameter Optimization
# Day 4

## Hydrological Modeling Using the Variable Infiltration Capacity Model (VIC) to Support Streamflow Monitoring in the Rufiji and Wami Ruvu Basins

*Andi Thomas*

*Regional Science Associate*

*SERVIR Eastern & Southern Africa*

# Overview for the Day

| Date/Time | Thursday 12th September |
|---|---|
| 8:30 – 9:00 | Arrival and Registration |
| 9:00 – 10:00 | Review / Model Results and Interpretation |
| 10:00 – 11:00 | Intro to Calibration Methods |
| 11:15 – 12:00 | Health Break |
| 12:00 – 12:30 | Model Calibration |
| 12:30 – 14:00 | Lunch Break |
| 14:00 – 15:00 | Model Calibration |
| 15:00 – 15:10 | Health Break |
| 15:10 – 16:10 | Model Calibration |

# Review

VIC is a gridded large-scale hydrologic model

VIC coupled with a routing model outputs streamflow for a given watershed →→→

Do we want to walk through the VIC documentation again?

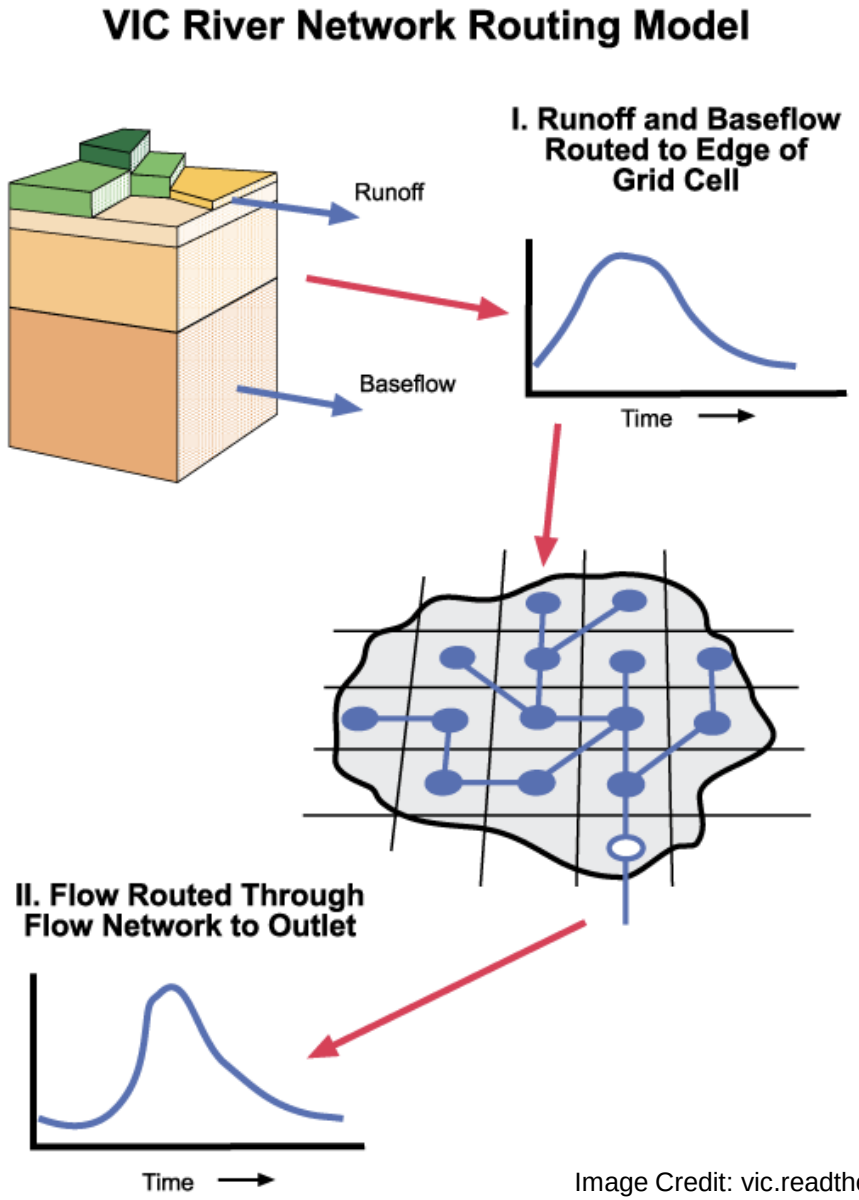



Image Credit: vic.readthedocs.io

# Why do we calibrate?

"All models are wrong, but some models are useful"

- George Box, Statistician

- Calibrating can improve estimations
- Decreases the gap/ inconsistencies between estimation and observation
- An increase in parameter heterogeneity in space/time = decrease in large-scale estimation accuracy
- It is easier to remove consistent biases. With further calibration, and removing bias, your various measures of error will improve. Yay!

# Common Measures of Error

## Nash–Sutcliffe model efficiency coefficient (NSE)

$$NSE = 1 - \frac{\sum_{t=1}^{T} (Q_m^t - Q_o^t)^2}{\sum_{t=1}^{T} \left(Q_o^t - \overline{Q_o}\right)^2}$$

Image Credit: Wikipedia

$Q_o$ – Average of the observed discharge
$Q_m$ – Modelled discharge
$Q_o^t$ – Modelled discharge at a given time (t)

NSE can range from -∞ to 1
1 – The model is perfect!
0 – model is as accurate as the mean
NSE < 0 – the average is better than the model
0.5 < NSE < 0.65 – model is good quality

## Root Mean Square Error (RMSE)

## R squared ($R^2$)

# Common Measures of Error

## Nash–Sutcliffe model efficiency coefficient (NSE)

$$NSE = 1 - \frac{\sum_{t=1}^{T} (Q_m^t - Q_o^t)^2}{\sum_{t=1}^{T} \left(Q_o^t - \overline{Q_o}\right)^2}$$

Image Credit: Wikipedia

$Q_o$ – Average of the observed discharge
$Q_m$ – Modelled discharge
$Q_o^t$ – Modelled discharge at a given time (t)

NSE can range from -∞ to 1
1 – The model is perfect!
0 – model is as accurate as the mean
NSE < 0 – the average is better than the model
0.5 < NSE < 0.65 – model is good quality

## Root Mean Square Error (RMSE)

Absolute measure of fit
Compares estimated and observed

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (P_i - O_i)^2}{n}}$$

Image Credit: Statistics How To

$P_i$ = estimated values
$O_i$ = observed values
n = the sum of observed minus estimated

RMSE range is defined by dependent variable
Smaller RMSE = better estimated values

## R squared ($R^2$)

# Common Measures of Error

## Nash–Sutcliffe model efficiency coefficient (NSE)

$$NSE = 1 - \frac{\sum_{t=1}^{T} (Q_m^t - Q_o^t)^2}{\sum_{t=1}^{T} \left(Q_o^t - \overline{Q_o}\right)^2}$$

Image Credit: Wikipedia

$Q_o$ – Average of the observed discharge
$Q_m$ – Modelled discharge
$Q_o^t$ – Modelled discharge at a given time (t)

NSE can range from -∞ to 1
1 – The model is perfect!
0 – model is as accurate as the mean
NSE < 0 – the average is better than the model
0.5 < NSE < 0.65 – model is good quality

## Root Mean Square Error (RMSE)

Absolute measure of fit
Compares estimated and observed

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} (P_i - O_i)^2}{n}}$$

Image Credit: Statistics How To

$P_i$ = estimated values
$O_i$ = observed values
n = the sum of observed minus estimated

RMSE range is defined by dependent variable
Smaller RMSE = better estimated values

## R squared ($R^2$)

Measurement of how well the estimated values match the observed values

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y})^2}$$

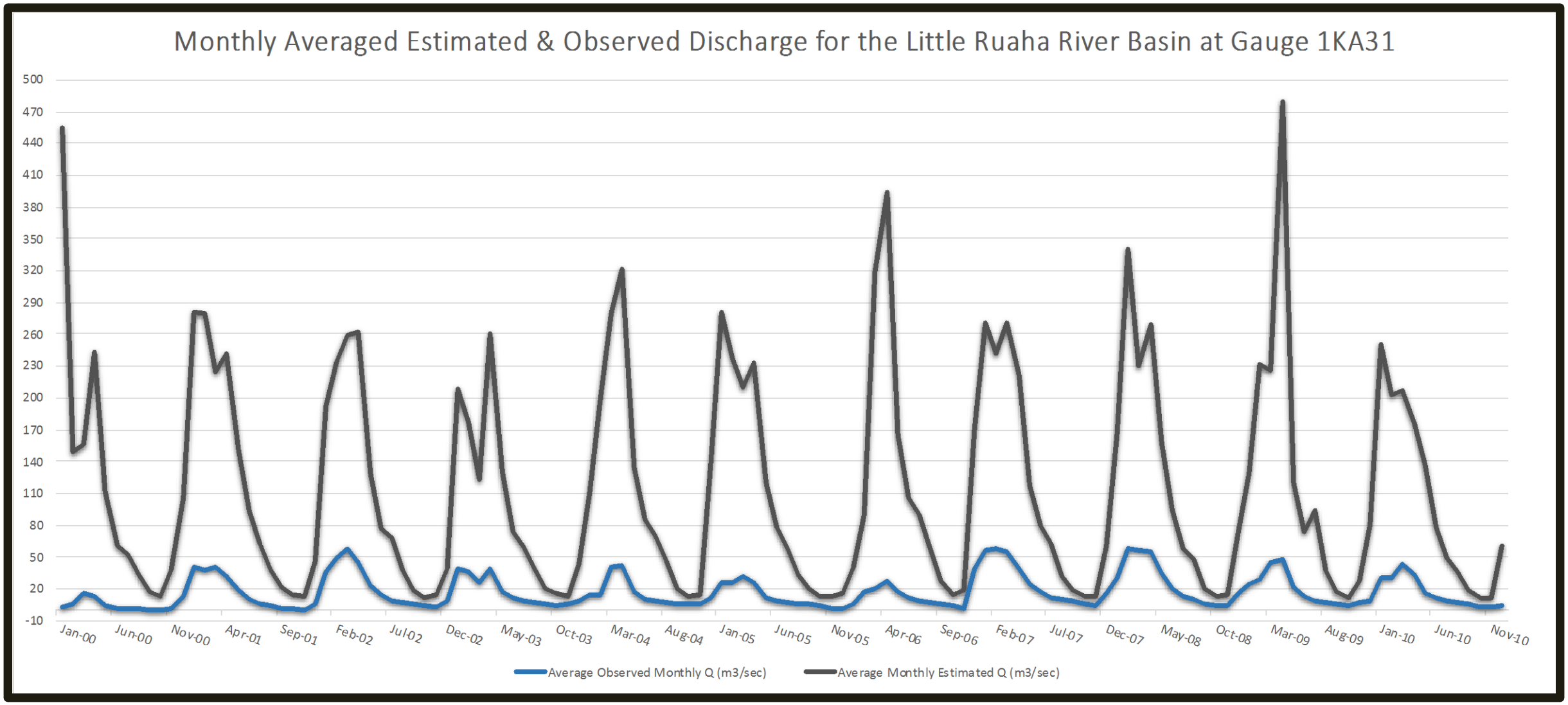
Image Credit: Wikipedia

$SS_{RES}$ = Residual sum of square error
$SS_{TOT}$ = Total sum of squared error
$y_i$ = Estimated values
$\overline{y}$ = Mean of predicted values
$y_i$ = Predicted values

Coefficient of Determination

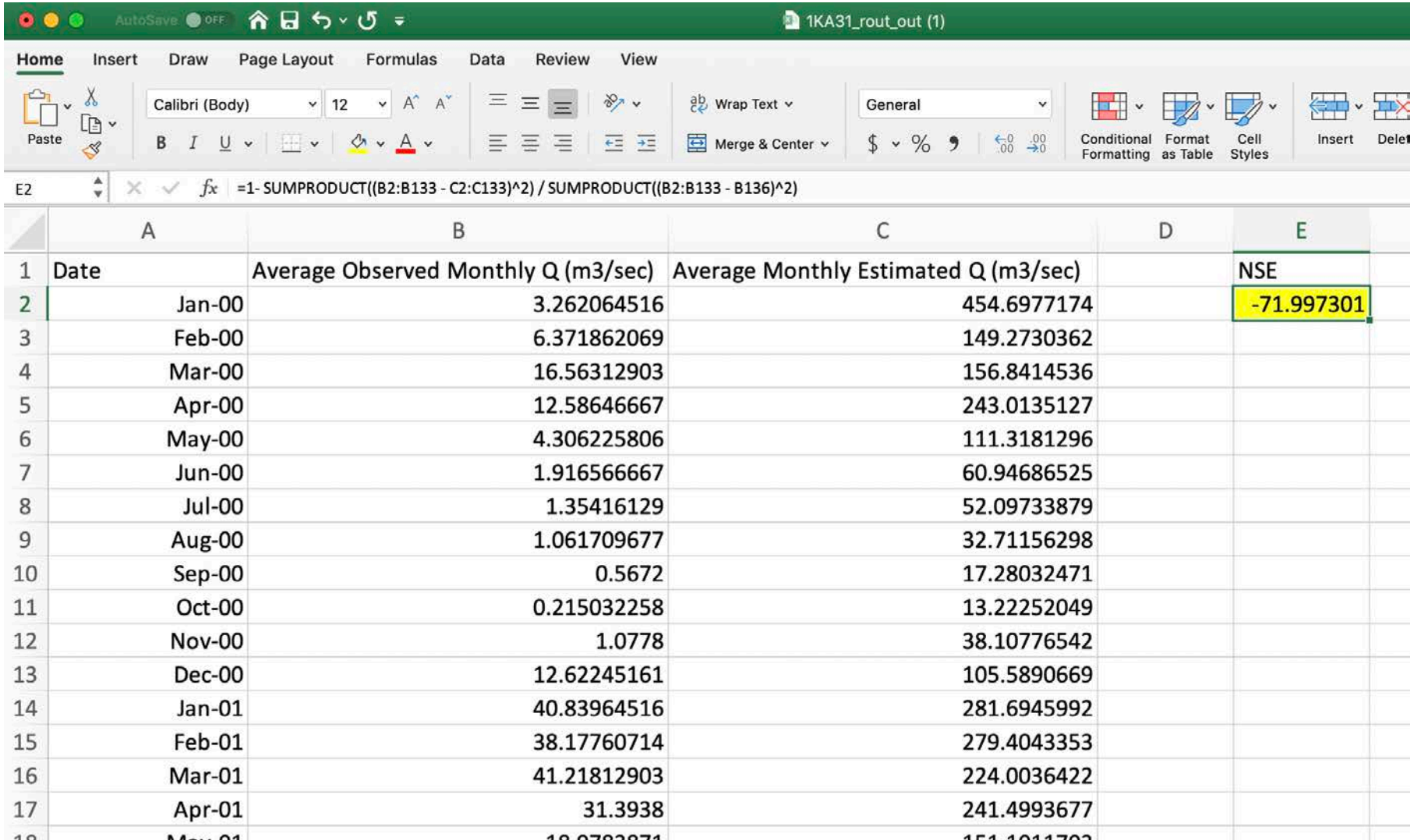
Values range from 0 – 1 (0 – 100%)

# Outputs from yesterday's VIC run



# Calculate NSE Demo

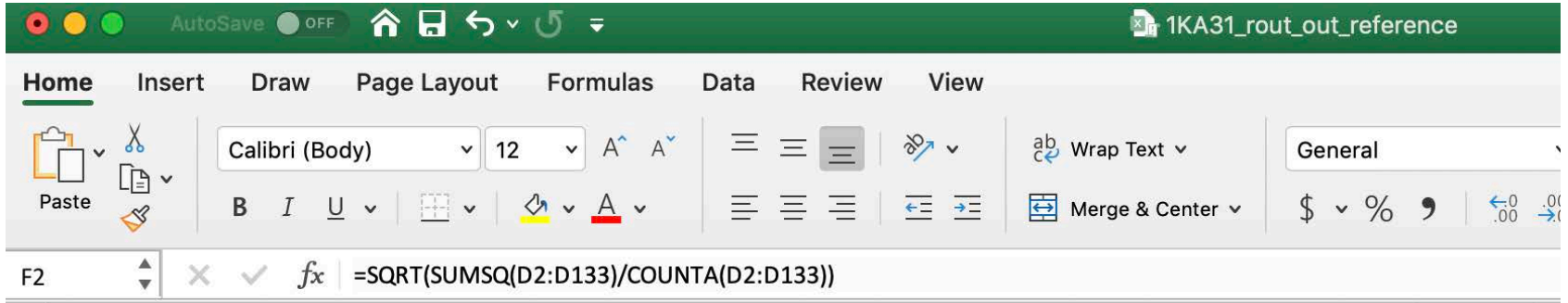
**1-SUMPRODUCT((Average Observed Q – Average Estimated Q)^2 / SUMPRODUCT((Average Observed Q – Average All Observed Years)^2)**




| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | Date | Average Observed Monthly Q (m3/sec) | Average Monthly Estimated Q (m3/sec) | | NSE |
| 2 | Jan-00 | 3.262064516 | 454.6977174 | | -71.997301 |
| 3 | Feb-00 | 6.371862069 | 149.2730362 | | |
| 4 | Mar-00 | 16.56312903 | 156.8414536 | | |
| 5 | Apr-00 | 12.58646667 | 243.0135127 | | |
| 6 | May-00 | 4.306225806 | 111.3181296 | | |
| 7 | Jun-00 | 1.916566667 | 60.94686525 | | |
| 8 | Jul-00 | 1.35416129 | 52.09733879 | | |
| 9 | Aug-00 | 1.061709677 | 32.71156298 | | |
| 10 | Sep-00 | 0.5672 | 17.28032471 | | |
| 11 | Oct-00 | 0.215032258 | 13.22252049 | | |
| 12 | Nov-00 | 1.0778 | 38.10776542 | | |
| 13 | Dec-00 | 12.62245161 | 105.5890669 | | |
| 14 | Jan-01 | 40.83964516 | 281.6945992 | | |
| 15 | Feb-01 | 38.17760714 | 279.4043353 | | |
| 16 | Mar-01 | 41.21812903 | 224.0036422 | | |
| 17 | Apr-01 | 31.3938 | 241.4993677 | | |

# Calculate RMSE Demo

**SQRT(SUMSQ ( Observed Q – Estimated Q ) / COUNTA (Observed Q – Estimated Q))**




| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | Date | Average Observed Monthly Q (m3/sec) | Average Monthly Estimated Q (m3/sec) | Observed - Estimated | | RMSE |
| 2 | Jan-00 | 3.262064516 | 454.6977174 | -451.4356529 | | 134.496048 |
| 3 | Feb-00 | 6.371862069 | 149.2730362 | -142.9011742 | | |
| 4 | Mar-00 | 16.56312903 | 156.8414536 | -140.2783245 | | |
| 5 | Apr-00 | 12.58646667 | 243.0135127 | -230.427046 | | |
| 6 | May-00 | 4.306225806 | 111.3181296 | -107.0119038 | | |
| 7 | Jun-00 | 1.916566667 | 60.94686525 | -59.03029858 | | |
| 8 | Jul-00 | 1.35416129 | 52.09733879 | -50.7431775 | | |
| 9 | Aug-00 | 1.061709677 | 32.71156298 | -31.6498533 | | |
| 10 | Sep-00 | 0.5672 | 17.28032471 | -16.71312471 | | |
| 11 | Oct-00 | 0.215032258 | 13.22252049 | -13.00748823 | | |
| 12 | Nov-00 | 1.0778 | 38.10776542 | -37.02996542 | | |
| 13 | Dec-00 | 12.62245161 | 105.5890669 | -92.96661527 | | |
| 14 | Jan-01 | 40.83964516 | 281.6945992 | -240.854954 | | |
| 15 | Feb-01 | 38.17760714 | 279.4043353 | -241.2267282 | | |
| 16 | Mar-01 | 41.21812903 | 224.0036422 | -182.7855132 | | |
| 17 | Apr-01 | 31.3938 | 241.4993677 | -210.1055677 | | |
| 18 | May-01 | 18.9783871 | 151.1011703 | -132.1227832 | | |
| 19 | Jun-01 | 10.62326667 | 93.10519346 | -82.48192679 | | |
| 20 | Jul-01 | 6.197129032 | 62.34453742 | -56.14740839 | | |
| 21 | Aug-01 | 3.799129032 | 37.3843843 | -33.58525527 | | |
| 22 | Sep-01 | 2.089166667 | 21.9054759 | -19.81630923 | | |
| 23 | Oct-01 | 1.100774194 | 13.87488453 | -12.77411034 | | |

# Computer-Based Calibration Methods

## *Computer-based calibrations:*

- Processing heavy
- Time-consuming
- Complex to learn

## *Manual Calibrations:*

- More time consuming for the user
- May not provide the best parameters
- User makes parameter changes based on extensive knowledge of the region

## Examples:

- ❖ Random Autostart Simplex Method
- ❖ Genetic Optimization Scheme
- ❖ Multi Objective COMplex Evolution
- ❖ Shuffled Complex Evolution

*These algorithms have the same goal to sync estimated and observed discharge for the period of record*

# Calibration Methods

*These algorithms have the same goal to sync estimated and observed discharge for the period of record*

## Random autostart simplex method

1. Selects a large number of random parameter sets at different grid cells and solves the model
2. The best set of parameters are selected from the random generation and used in the simplex minimization algorithm
3. The simplex minimization algorithm groups the minimum values in a geometric shape (the simplex), with N + 1 apexes
    - N is the number of parameters that are being optimized
4. With the minimum, the algorithm minimizes the volume of the simplex until all of the parameters are within a tolerance of each other
5. The process restarts continuously until until the *global optimization parameters* are located

Limitation: The algorithm will *first find the local minimum instead of the global minimum* for model optimization. If the the algorithm “re-starts” enough times the global optimization parameters will eventually be reached

# Global Minimum vs. Local Minimum

We are not getting the whole picture…

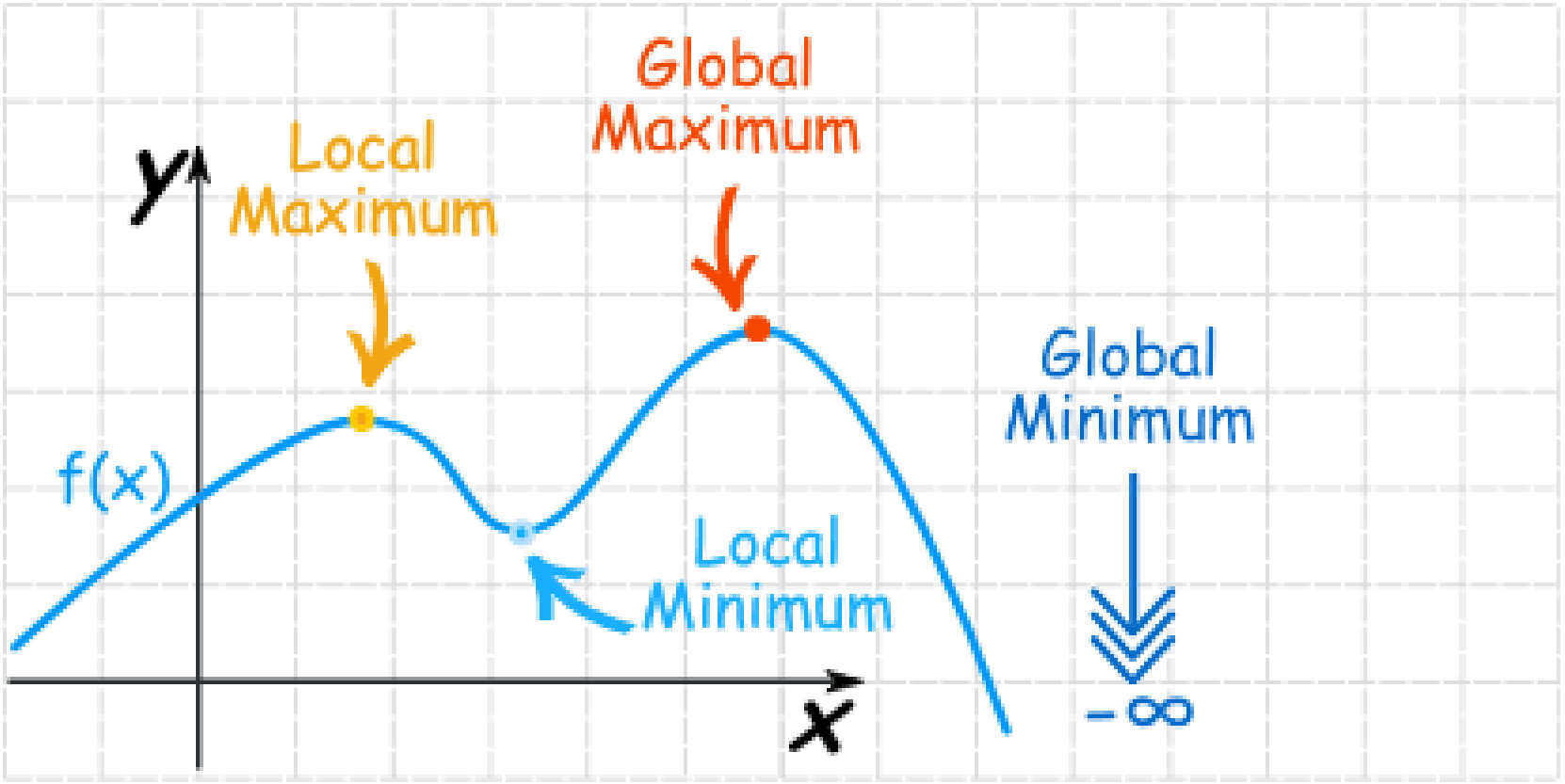



Image Credit: Math is fun

# Calibration Methods

*These algorithms have the same goal to sync estimated and observed discharge for the period of record*

## Genetic optimization scheme (*survival of the fittest)*

1. A random set of parameters is generated
2. The parameters are sorted and assigned a probability of reproducing
   -lowest $-R^2$ values are assigned highest probabilities
3. A new generation is created by randomly selecting parameters from the previous generation
4. The parameter sets for both generations are encoded as bit strings
5. Fragments are selected from both parameter sets and recombined to form a new set
6. The model is solved for the new set of parameters
7. The process repeats until the less optimal parameters are cycled out
   -Eventually the algorithm provides optimized parameters

**Tested by the University of Washington and failed after 6 days! Moving on…**

# Calibration Methods

*These algorithms have the same goal to sync estimated and observed discharge for the period of record*

## Multi Objective COMplex Evolution (MOCOM-UA) (*Yapo, et al., 1998*)

- Pareto principle: a theory that says 80% of the output from a system is determined by 20% of the input
- This calibration method produces optimized parameters that define a subset of the Pareto and increases the number of simulations.

1. Model selects a randomly distributed number of observations throughout the parameter space; a uniform sampling distribution is used
2. For each sample the multi-objective vector is computed and the observations are ranked and sorted using the Pareto-ranking procedure
3. Simplexes of *s+1* observations are selected according to the robust rank-based selection method
4. A multi-objective extension of the downhill simplex method is used to evolve each simplex in a multi-objective improvement direction
5. Iterative application of the ranking and evolution procedures causes all observations to converge towards the Pareto optimum (global optimum parameters)

# Calibration Methods: IN OTHER WORDS

*These algorithms have the same goal to sync estimated and observed discharge for the period of record*

## Multi Objective COMplex Evolution (MOCOM-UA) (*Yapo, et al., 1998*

*In terms of natural selection…*

1. Think of the parameter values as rabbits in a population
2. Each rabbit is a potential parent with the ability to reproduce baby rabbits
3. A selected simplex is like a set of three parent rabbits
4. The "better" parents have a higher probability of getting to contribute to the offspring population than "lesser" parents (The selection of three parents makes the scenario a competitive one)
5. The same multi-objective strategy is applied to each group of parents to generate offspring
6. Each new offspring replaces the "lesser" offspring and each parent gets a chance at reproducing before being discarded; Every rabbit gets a chance!

**Please explore this algorithm on your own time. Example can be found here:**

**https://vic.readthedocs.io/en/vic.4.2.d/Documentation/MOCOM/**

# Calibration Methods

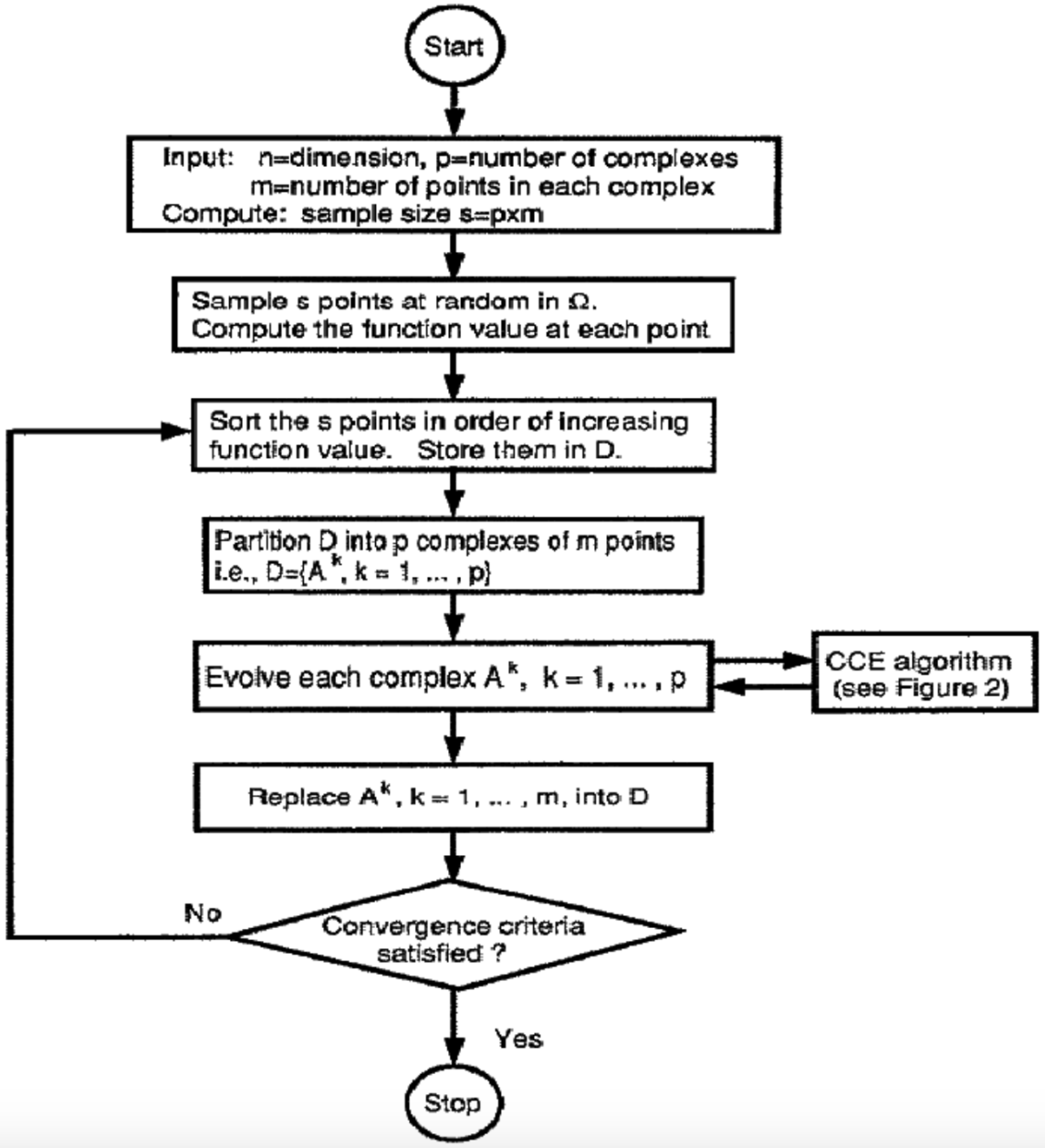
*These algorithms have the same goal to sync estimated and observed discharge for the period of record*

## Shuffled Complex Evolution (SCE-UA)(Duan et al.)

*A very common calibration method*

1. A random sample of parameter values is created using a uniform probability distribution and a criterion value is calculated for each parameter value
2. The parameter values are sorted  and ranked based on the criterion value-– the smallest value is first and the largest criterion value is last
3. The parameter values are then put into groups
4. Each group evolves according to the Competitive Complex Evolution (CCE) algorithm
5. Groups are re-split and shuffled around
6. If any of the new values meet the global optimization parameters (convergence) the algorithm will stop (otherwise, the process repeats itself)

# SCE-UA Flow Chart (Duan et al.)



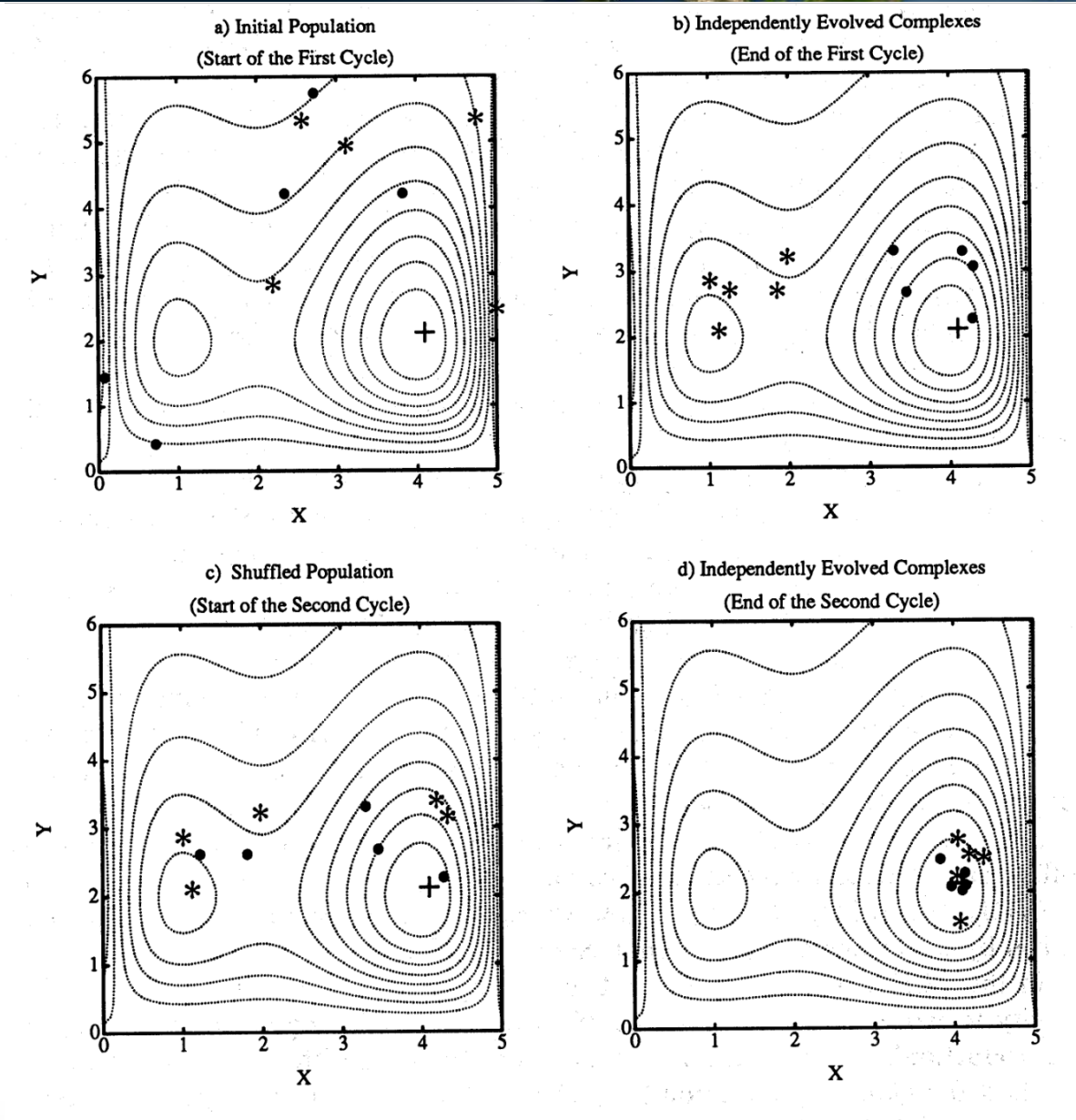




Fig. 1. Illustration of the shuffled complex evolution (SCE-UA) method.

# Common Parameters to Calibrate

**b_infilt** | Ws | Ds | Dsmax | soil depth

b_infilt is the Variable Infiltration Curve

Variable Infiltration Curve values range from 10.5 to 0.4

Higher values will produce more runoff

0.2 is the default starting value

A higher value of $b_{inf}$ gives lower infiltration and yields higher surface runoff

# Common Parameters to Calibrate

b_infilt | **Ws** | Ds | Dsmax | soil depth

Ws is the fraction of maximum soil moisture where non-linear baseflow occurs.

Ws values are usually greater than 0.5

0.9 is the default starting value

A higher value of Ws will raise the water content required for rapidly increasing, non-linear baseflow, which will tend to delay runoff peaks

# Common Parameters to Calibrate

b_infilt | Ws | **Ds** | **Dsmax** | soil depth

Dsmax is the maximum velocity of baseflow for each grid cell in mm/day

*Dsmax = saturated hydraulic conductivity (Ksat) at each grid cell * slope of grid cell*

Ds is the fraction of Dsmax where non-linear baseflow occurs in millimeters

With a higher value of Ds, the baseflow will be higher at lower water content in lowest soil layer

# Common Parameters to Calibrate

b_infilt | Ws | Ds | Dsmax | **soil depth**

soil depth (depth) is the depth of each soil layer in meters.

Typical values for each layer can range from 0.1 – 1.5 meters

Soil depth effects many model variables

- For runoff considerations, thicker soil depths slow down (baseflow dominated) seasonal peak flows and increase the loss due to evapotranspiration

# Calibration Tips & Tricks

- No universal method of calibration exists for all models
- Globally calibrated models perform well globally (not locally)
- **Take the split-sample approach: Set aside part of the period of record to validate your calibrated dataset (approx. half)**
- Vegetation classes and snow bands are computationally expensive!
  - -Cells with <1-2% of the vegetation classes can be removed
- Calibrate only “wet” cells that contribute to 75% of basin’s streamflow
- Aggregate cells to 1 degree if you are working with lower resolutions
  - -Once you have targeted parameters that yield good results you can extract to a higher resolution

# calibrate_vic_sceua.py

*Navigate to your scripts folder*

**We need to modify/check:**

1. Validation time series path
2. File paths for parameter and GIS files
3. Output paths
4. Routing variables
5. Simulation time series
6. Calibration start and end dates
7. Calibration output file

```
27  class vic_model(object):
28
29      def __init__(self,startTime,endTime):
30          self.st = startTime
31          self.et = endTime
32
33          return
34
35      def get_obs(self):
36          # specify validation time series path
37          valFile = '../input/Nyando_discharge.xlsx'
38
39          obsData = pd.ExcelFile(valFile)
40          obsSheet = obsData.parse('Daily')
41          obstimes = pd.date_range('2005-02-01','2013-12-31',freq='D')
42          obsSeries = xr.DataArray(obsSheet.Q,coords=[obstimes],dims=['time']).se
43
44          self.observations = obsSeries
45
46          return
47
48      def run_vic(self,binfilt=None,Ws=None,Ds=None,c=None,soil_d2=None,soil_d3=N
49
50          __location__ = os.path.realpath(
51          os.path.join(os.getcwd(), os.path.dirname(__file__)))
52
53          os.chdir(__location__)
54
55          # specify file paths to pass into system commands
56          globalFile = os.path.join(__location__,'../input/global.params')
```

# Calibration Steps Using SCE-UA for 1KA31

*We will calibrate our estimated VIC outputs for the Little Ruaha River (at 1KA31) with CMORPH data*

1. Point the Global Parameter file to the CHIRPS precipitation dataset by changing the file path for the meteorological forcings row.

2. Open terminal and navigate to the “scripts” folder within the training directory

3. Run command: **`$ python calibrate_vic_sceua.py 10`**

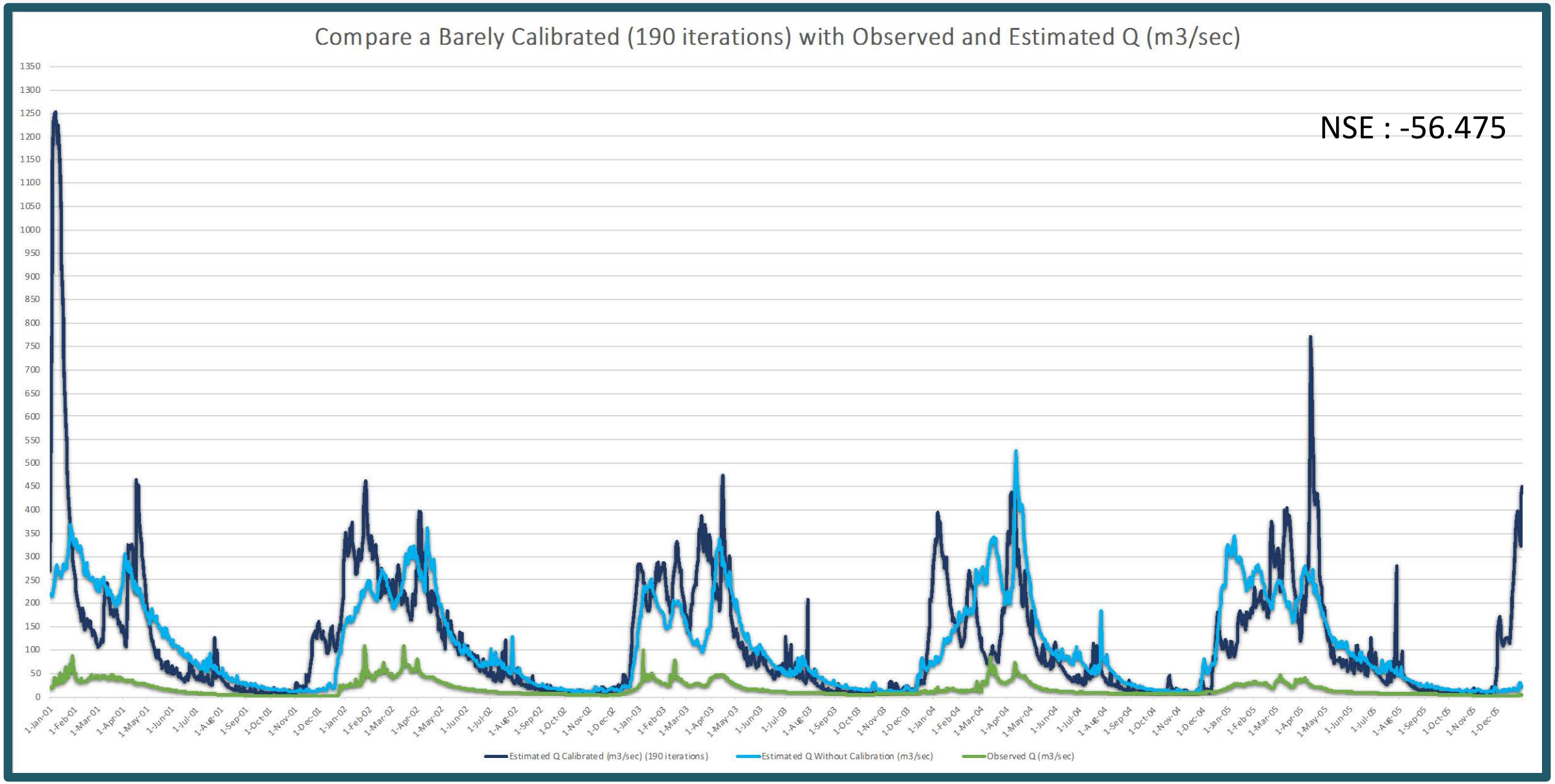
   -10 is the number of iterations we will run

   -The number of iterations you choose for full calibration should be 5,000 – 10,000 (10,000 is a loose standard)

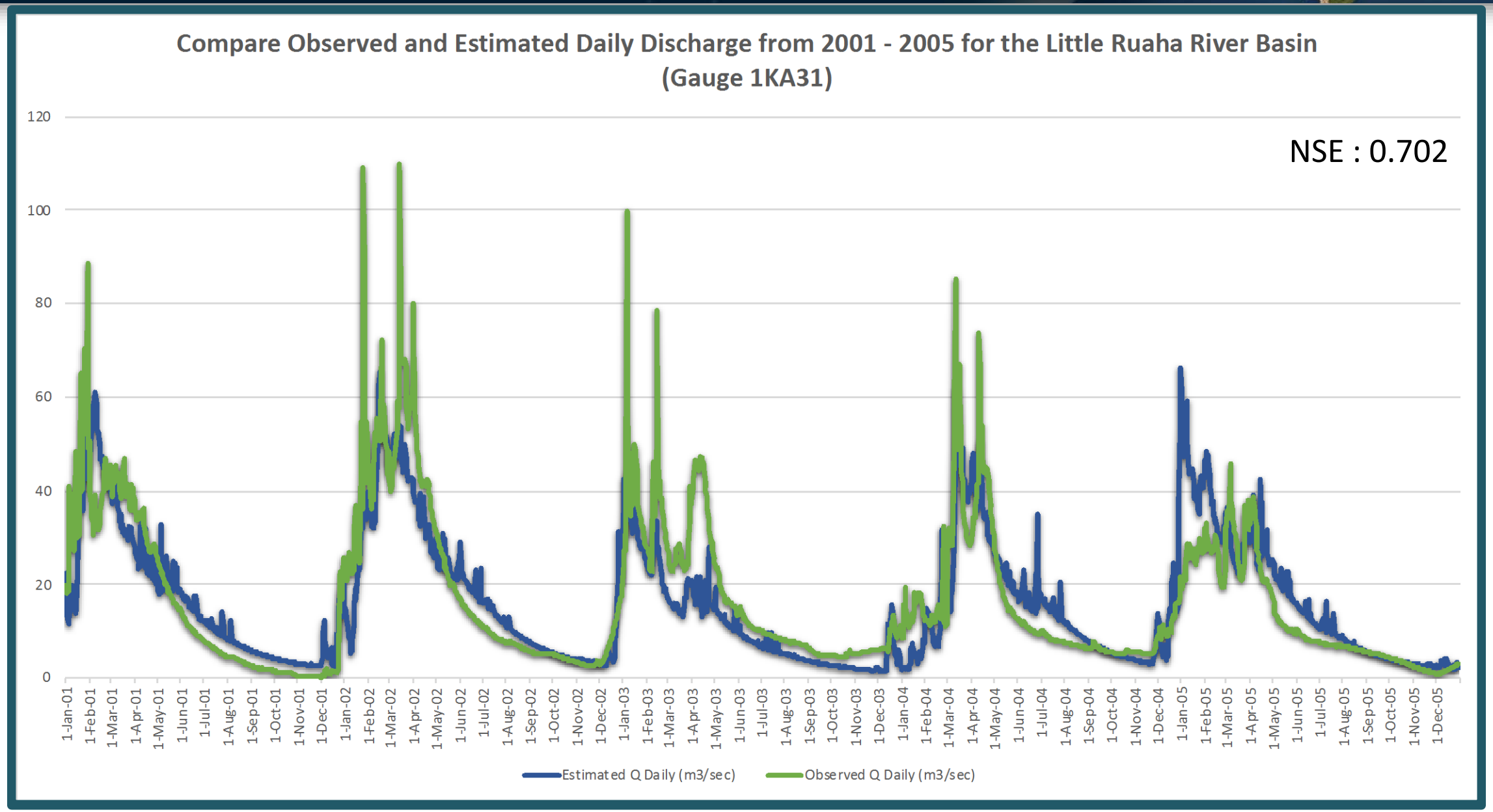
   -This will take a while (maybe more than a week for 10,000 iterations!)

   -Algorithm uses the Nash-Sutcliffe coefficient as the calibration objective function

# Results after 190 iterations…



# Results after 6,358 iterations…



# What if the calibration script stops?

Quick Fix:

1. Open calibrated discharge file (SCEUA_VIC_1KA31.csv)
2. Sort “like1” column from *greatest to least* (this is your NSE value)
3. Copy parameter values on second row somewhere (text file, notepad, etc.)
    -parbinfil (Variable Infiltration Curve)
    -parWs (fraction of max soil moisture where non-linear baseflow occurs)
    -parSoilD2 (Soil Layer One)
    -parSoilD3 (Soil Layer Two)
4. Open format_soil_params.py

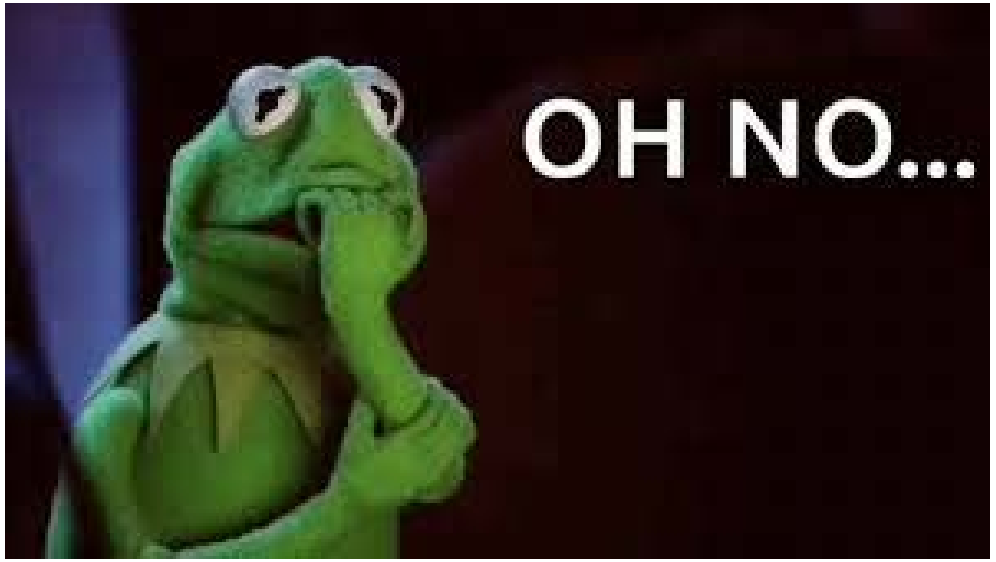



Image Credit: Oh No GIFs, Tenor, Tenor.com

# What if the calibration script stops?

## Original Script

```
159             if b_val == None:
160                 b_val = soilDrain['infilt']
161             if Ds_val == None:
162                 Ds_val = soilDrain['Ds']
163             if Ws_val == None:
164                 Ws_val = soilDrain['Ws']
165             if s2 == None:
166                 s2 = 1.50
167             if s3 == None:
168                 s3 = 0.30
169
170             grdc = cnt
171             lat = yy[i,j]
172             lon = xx[i,j]
173             infilt = b_val
174             Ds = Ds_val
175             Dsmax = (data[i,j,4]/100.) * (float(subSoilPro['S
176             Ws = Ws_val
177             c = 2
178             expt = 3+(2*float(topSoilPro['SlopeRCurve']))
179             expt1 = 3+(2*float(subSoilPro['SlopeRCurve']))
180             tksat = (float(topSoilPro['SatHydraulicCapacity']
181             sksat = (float(subSoilPro['SatHydraulicCapacity']
182             phis = -999
183             elev = data[i,j,2]
184             depth = 0.10
185             depth1 =s2
186             depth2 =s3
187             avg_t = 27
```

## Modified Script

```
159             #if b_val == None:
160                 #b_val = soilDrain['infilt']
161             #if Ds_val == None:
162                 #Ds_val = soilDrain['Ds']
163             #if Ws_val == None:
164                 #Ws_val = soilDrain['Ws']
165             #if s2 == None:
166                 #s2 = 1.50
167             #if s3 == None:
168                 #s3 = 0.30
169
170             grdc = cnt
171             lat = yy[i,j]
172             lon = xx[i,j]
173             infilt = 0.37305
174             Ds = 0.13965
175             Dsmax = (data[i,j,4]/100.) * (float(subSoilPro['S
176             Ws = 0.0578
177             c = 2
178             expt = 3+(2*float(topSoilPro['SlopeRCurve']))
179             expt1 = 3+(2*float(subSoilPro['SlopeRCurve']))
180             tksat = (float(topSoilPro['SatHydraulicCapacity']
181             sksat = (float(subSoilPro['SatHydraulicCapacity']
182             phis = -999
183             elev = data[i,j,2]
184             depth = 0.10
185             depth1 = 0.7583
186             depth2 = 0.99805
187             avg_t = 27
```

# Additional Resources

- VIC Read the Docs: https://vic.readthedocs.io/en/master/
  - Calibration Methods: https://vic.readthedocs.io/en/vic.4.2.d/Documentation/Calibration/
- Spotpy: https://pypi.org/project/spotpy/
- Python 3 Course: https://www.codecademy.com/learn/learn-python-3
- RVIC: https://rvic.readthedocs.io/en/latest/

# References


Duan, Qingyun & Sorooshian, Soroosh & Gupta, Vijai. (1992). Effective and Efficient Global Optimization for Conceptual Rainfall-Runoff Models. Water Resources Research - WATER RESOUR RES. 28. 1015-1031. 10.1029/91WR02985.

Liang, X., D. P. Lettenmaier, E. F. Wood, & S. J. Burges (1994), A simple hydrologically based model of land surface water and energy fluxes for general circulation models, *J. Geophys. Res.*, 99, 14415-14428

Yapo, P., H. Gupta, and S. Sorooshian, 1998: Multi-objective global optimization for hydrologic models, *J. Hydrology*, **204**(1), 83-97.