# Professional Paper - Chapters 1,2, and 3

William Cook

## 0.1 Abstract

Dynamic models that simulate processes across large geographic locations, such as hydrologic models, are often informed by spatially distributed parameters. Spatially distributed parameters are frequently correlated and any techniques utilized in their calibration ideally incorporate existing hierarchical relationships into their structure. In this paper, a parameter estimation method based on the Dual State Ensemble Kalman Filter called the Dual State Hierarchical Ensemble Kalman Filter (DEHnKF) is presented. This modified filter is innovative in that it allows parameters to be placed into a series of groups that are smoothed using hierarchical modeling techniques. The usability and effectiveness of this new technique is demonstrated by applying it to *daWUAPhydroengine*, a rainfall-runoff model that simulates subcatchment-scale hydrologic processes and contains high dimensional spatially distributed empirical parameters.

# Contents

# Chapter 1

# Introduction

Utilizing sequential data assimilation techniques to filter hydrologic models is an efficient way to correct and calibrate them both before and after implementation in the field. Observations such as SWE (snow water equivalent), streamflow, and precipitation are collected on a daily basis across various geographic regions, allowing real time information to be dynamically ingested by the hydrologic model and inform present and future predictions. More accurate models allow hydrologists to better understand the past and predict the future, and the need to research optimal methods of hydrologic data assimilation has been recognized [18] and researched [10], [15]. Observed hydrologic data may allow models, including rainfall-runoff models, to undergo parameter estimation. Parameter estimation for rainfall-runoff models has been a relevant field of research for quite a while [16],[17] and research has progressed into the 21'st century [13], [20].

Models that ingest data sequentially can have their parameters efficiently corrected by a Kalman Filter, a sequential data assimilation algorithm. Kalman Filters only need the previous timestep's state estimate, parameter estimate, and co-variance matrices to update the current timestep's state estimate, parameter estimate, and co-variance matrices. The original Kalman filter[8] was created to solve linear problems and more complicated implementations must be used to solve non-linear problems. The extended Kalman Filter[6] works for mildly non-linear systems but does not function optimally on heavily non-linear systems[12]. The Unscented Kalman Filter[7]

is an all-around improvement on the Extended Kalman Filter that allows for the filtering of highly non-linear systems. The Ensemble Kalman Filter[4], a predecessor to the Unscented Kalman Filter, filters non-linear systems by generating an 'ensemble' of model instances and adding unique noise to each model's forcing data. The main advantage of this ensemble based approach is the substitution of the original Kalman Filter's error covarience matrix with an ensemble covariance matrix, which allows for the efficient computation of the covariance of high dimensional state vectors.

To calibrate model parameters as well as model states a Dual State Kalman Filter may be used as demonstrated by Moradkhani et. al in 2005 [13]. Dual state Kalman filters add a small perturbation to a series of parameters that the user wishes to calibrate. These perturbed parameters vectors are then corrected in a similar fashion to the state vectors. After this happens a second filter is run to correct the state vectors in the traditional fashion. The Dual State Ensemble Kalman Filter implemented by Moradkhani et. al[13] extends the Ensemble Kalman Filter into a dual state configuration and is shown to successfully predict a set of parameters.

An alternative method of parameter estimation that utilizes the Kalman Filter is the Joint Kalman Filter, which combines states and parameters into one vector that is calculated simultaneously without the need for a second run. Joint Ensemble Kalman Filters have been successfully implemented on hydrologic models [19] and other models [3], but Joint Ensemble Kalman filters can suffer from "filter inbreeding" under certain circumstances [5] and introduce inconsistency in especially heterogeneous formations [21]. Overall, Dual Ensemble Kalman Filters are more accurate then Joint Ensemble Kalman Filters, especially in noisy situations, with the major drawback of the Dual approach being its larger draw on computational power [11].

In this paper hierarchical modeling techniques are integrated into the Dual State Ensemble Kalman Filter's parameter perturbation equation to create a Hierarchical Dual State Ensemble Kalman Filter. A hierarchical parameter perturbation framework allows the model to account for parameters that are hierarchically related. To examine the Dual State Hierarchical Ensemble Kalman Filter's application to high dimensional spatially distributed raster data and geographical data *daWUAPhydro-*

*engine*, a variation of rainfall-runoff model, is implemented to predict streamflows across the state of Montana. *daWUAPhydroengine* is informed by a variety of sub-components featuring high dimensional spatially distributed parameters, including a snowpack process, soil process, and a Muskingham-Cunge routing component. Conveniently, these parameters can be linked to individual sub-basins with can in turn be sorted into HUC-4 class watersheds, which calls for a hierarchical approach. Despite the model's high-dimensional parameters, *daWUAPhydroengine* is designed to be a quick and efficient model. Accordingly, a Dual State Hierarchical Ensemble Kalman Filter is an optimal calibration algorithm to calibrate this raster data because 1) the DEHnKF does not have to compute the high dimensional state covariance matrix and 2) *daWUAPhydroengine* is a sequential model that could conceivably benefit from real-time parameter correction.

Chapter 2 covers the methods behind the Dual State Hierarchical Ensemble Kalman Filtering algorithm. Chapter 3 discusses *daWUAPhydroengine* and how a Dual State Hierarchical Ensemble Kalman Filter was applied to it. Chapter 4 discusses results while Chapter 5 compares those results with calibrated parameters from a Dual State Ensemble Kalman Filter as implemented by Moradkhani et al.

# Chapter 2

# The Hierarchical Dual Ensemble Kalman Filter Method

## 2.1 General Dynamic Model and Observations

A generic dynamic model can be defined as one more more discrete nonlinear stochastic processes[3]:

$$x_{t+1} = f(x_t, u_t, \theta_t) + \varepsilon_t \tag{2.1}$$

where $x_t$ is an $n$ dimensional vector representing the state variables of the model at time step $t$, $u_t$ is a vector of forcing data (e.g temperature or precipitation) at time step $t$, and $\theta_t$ is a vector of model parameters which may or may not change per time step (e.g *soil beta* or $DDF$). The non-linear function $f$ takes these variables as inputs. The noise variable $\varepsilon_t$ accounts for both model structural error and for any uncertainty in the forcing data.

A state's observation vector $z_t$ can be defined as

$$z_t = h(x_t, \theta_t) + \delta_t \tag{2.2}$$

Where the $x_t$ vector represents the true state, $\theta_t$ represents the true parameters, $h(.)$ is a function that determines the relationship between observation and state

vectors, and $\delta_t$ represents observation error. $\delta_t$ is Gaussian and independent of $\varepsilon_t$.

The Dual Hierarchical State Ensemble Kalman Filter can be split into three subsections: The prediction phase, the parameter correction phase, and the state correction phase.

## 2.2 DEnHKF Method

### 2.2.1 Prediction Phase

Just as in a standard Dual Ensemble Kalman filter (see **??** for the Dual Ensemble Kalman Filter methods as implemented by Moradkhani et al.), each ensemble member $i$ is represented by a stochastic model similar to (2.1). The modified equation is as follows:

$$x_{t+1}^{i-} = f(x_t^{i+}, u_t^i, \theta_t^{i-}) + \omega_t, \quad i = 1, ..., n \tag{2.3}$$

Where $n$ is the total number of ensembles. The $-/+$ superscripts denote corrected $(+)$ and uncorrected $(-)$ values. Note that $\theta_t^{i-}$'s $t$ subscript does not necessarily denote that $\theta$ is time variant but rather indicates that parameter values change as they are filtered over time. The noise term $\omega_t$ accounts for model error and will hereafter be excluded from the state equation.

Errors in the model design are accounted for through the perturbation the forcing data vector $u_t$ with random noise $\zeta_t^i$ to generate a unique variable $u_t^i$ for each ensemble. $\zeta_t^i$ is drawn from a normal distribution with a covarience matrix $Q_t^i$.

$$u_{t+1}^i = u_t + \zeta_t^i, \quad \zeta_t^i \sim N(0, Q_t^i) \tag{2.4}$$

#### Hierarchical Parameter Perturbation

To generate the priori parameters $\theta_{t+1}^{i-}$ an evolution of the parameters similar to the evolution of the state variables must be implemented. Legacy implementations of parameter evolution added a small perturbation sampled from $N(0, \Sigma_t^\theta)$, where $\Sigma_t^\theta$

represents the covariance matrix of $\theta$ at timestep $t$. This legacy method of evolution resulted in overly dispersed parameter samples and the loss of continuity between two consecutive points in time [9] [3]. To overcome this the kernel smoothing technique developed by West [22] and implemented by Liu [9] has been used effectively in previous Dual Ensemble Kalman filter implementations [13] and similar models [3].

$$\theta_{t+1}^{i-} = a\theta_t^{i+} + (1-a)\bar{\theta}_t^{+} + \tau_t^i \tag{2.5}$$

$$\tau_t^i = N(0, h^2 V_t) \tag{2.6}$$

Where $\bar{\theta}_t^{+}$ is the mean of the parameters with respect to the ensembles, $V_t = var(\theta_t^{i+})$, $a$ is a shrinkage factor between (0,1) of the kernel location, and $h$ is a smoothing factor. $h$ may be defined as $\sqrt{1-a^2}$, while $a$ may fall between (.45,.49) [3]. Note that $h$ and $a$ tend to vary per model and optimal values for these parameters are generally found via experimentation [13] [1] [2] [3].

In a Hierarchical Duel Ensemble Kalman Filter, parameter perturbation is accomplished using a hierarchical algorithm. For a simple overview of hierarchical models refer to Osborne [14]. First, ensembles are placed into a series of groups $G_g$ based on shared characteristics (spatial or otherwise.) Algorithms (2.5) and (2.6) are then updated to conform to the hierarchical model structure:

$$\theta_{t+1}^{i-,g} = a\bar{\theta}_{t,g}^{+} + (1-a)\left\langle\theta\right\rangle_{t,g}^{i+} + \tau_{t,g}^i \tag{2.7}$$

$$\tau_{t,g}^i = N(0, h^2 V_{t,g}^i) \tag{2.8}$$

$$V_t^i = a * var(\theta_{t,g}) + (1-a)var(\theta_t^{i-}) \tag{2.9}$$

Where $a\bar{\theta}_{t,g}^{+}$ is the mean over all ensembles for all members of group $g$, $\left\langle\theta\right\rangle_{t,g}$ is the mean of all members in ensemble $i$, group $g$, $var(\theta_{t,g})$ is the variance over the ensembles for all members of group $g$, and $var(\theta_t^{i-})$ is the variance of all members in ensemble $i$, group $g$.

### 2.2.2 Parameter Correction Phase

In an Ensemble Kalman Filter, observations are perturbed to reflect model error. To accomplish this $n$ unique perturbations are created. Therefore, the variable $z^i_{t+1}$ is defined as follows:

$$z^i_{t+1} = z_{t+1} + \eta^i_{t+1}, \quad \eta^i_{t+1} = N(0, R_{t+1}) \tag{2.10}$$

Where $z_{t+1}$ is an observation vector defined by (2.2) and $\eta^i_{t+1}$ is a random perturbation drawn from a normal distribution with covarience matrix $R_{t+1}$. A set of state predictions that can be related to the observations are generated by running the priori state vector through the function $h(.)$:

$$\hat{y}^i_{t+1} = h(x^{i-}_{t+1}, \theta^{i-}_{t+1}) \tag{2.11}$$

The parameter update equation is similar to the update equation of the linear Kalman filter $\hat{x}^+_t = \hat{x}^-_t + K_t(z_t - H\hat{x}_t)$. Notably, parameters are corrected in lieu of the states:

$$\theta^{i+}_{t+1} = \theta^{i-}_{t+1} + K^\theta_{t+1}(z^i_{t+1} - \hat{y}^i_{t+1}) \tag{2.12}$$

To facilitate this, $K^\theta_{t+1}$ is defined as

$$K^\theta_{t+1} = \frac{\Sigma^{\theta,\hat{y}}_{t+1}}{\Sigma^{\hat{y},\hat{y}}_{t+1} + R_{t+1}} \tag{2.13}$$

where $\Sigma^{\theta,\hat{y}}_{t+1}$ is the cross covariance of $\theta_{t+1}$ and $\hat{y}_{t+1}$, $\Sigma^{\hat{y},\hat{y}}_{t+1}$ is the covarience of $\hat{y}_{t+1}$, and $R_{t+1}$ is the observation error matrix from (2.10).

### 2.2.3 State Correction Phase

After $\theta^{i+}_{t+1}$ has been calculated the model is run again (2.3) with the $\theta^{i+}_{t+1}$ replacing $\theta^{i-}_{t+1}$.

$$x_{t+1}^{i-} = f(x_t^{i+}, u_t^i, \theta_t^{i+}), \quad i = 1, ..., n \tag{2.14}$$

After a new state vector is generated it is re-run through (2.11) with the new parameter vector:

$$\hat{y}_{t+1}^i = h(x_{t+1}^{i-}, \theta_{t+1}^{i+}) \tag{2.15}$$

The corrected state vector is then run through the state update equation

$$x_{t+1}^{i+} = x_{t+1}^{i-} + K_{t+1}^x(z_{t+1}^i - \hat{y}_{t+1}^i) \tag{2.16}$$

$$K_{t+1}^x = \frac{\Sigma_{t+1}^{x,\hat{y}}}{\Sigma_{t+1}^{\hat{y},\hat{y}} + R_{t+1}} \tag{2.17}$$

where $\Sigma_{t+1}^{x,\hat{y}}$ is the cross covariance of $x_{t+1}$ and $\hat{y}_{t+1}$.

# Chapter 3

# Application of DEnHKF to Hydrologic Model

## 3.1 daWUAPhydroengine

The **daWUAPhydroengine** hydrologic dynamic model is used to test the viability of the DEnHKF method. **daWUAPhydroengine** takes streamflow and subbasin parameters, precipitation, minimum temperatures, and maximum temperatures as inputs and outputs streamflow data along with some additional states such as snow water equivalent. **daWUAPhydroengine** was designed to be implemented in any geographic location. For this study it was utilized to model streamflows throughout the state of Montana.

Configuring **daWUAPhydroengine** to model streamflows throughout Montana is advantageous because it allows for the calibration of a very large number of spatially distributed, high dimensional parameters. These parameters span the entirety of Montana, which covers an area of 380,800 $km^2$. Montana's large geographical coverage

Table 3.1: States

| State ($x$) | Purpose | Dimensions |
|---|---|---|
| streamflow | Streamflow (in cumecs) | 330 |
| swe | Snow Water Equivalent (in $mm^3$) | 45012 |

Table 3.2: Forcing Data

| Forcing Data ($u$) | Purpose | Dimensions |
|---|---|---|
| tempmin | Lowest temperature for timestep | 45012 |
| tempmax | Highest temperature for timestep | 45012 |
| precipitation | Amount of rainfall for timestep | 45012 |

Table 3.3: Calibrated Parameters

| Parameter ($\theta$) | Purpose | Dimensions |
|---|---|---|
| ddf | Controls Rate of Snowfall | 45012 |
| aet_lp | Controls AET | 45012 |
| soil_beta | Controls portion of ponded water that goes into soil storage | 45012 |
| soil_max_wat | Controls soil maximum water capacity | 45012 |

is diverse and the terrain differs in various ways (soil composition, forestation, etc.)

## 3.2 Observation Data

A Kalman Filter relies on one or more observed states for correction. Accordingly, observations were obtained for streamflows across Montana and snowfall across Montana. For streamflow, USGS streamflow data was collected at 86 sites. Each observed site was paired with the closest simulated **daWUAPhydroengine** stream outlet within a 2.5 mile cutoff. For snowfall, SNOWTEL sites monitored by the Natural Resources Conservation Service (NRCS) were used. 90 stations were chosen and matched to specific pixels in **daWUAPhydroengine**'s raster files.

Table 3.4: Observations

| Observed State ($x$) | Source | Dimensions |
|---|---|---|
| streamflow | USGS | 82 |
| swe | NRCS | 90 |

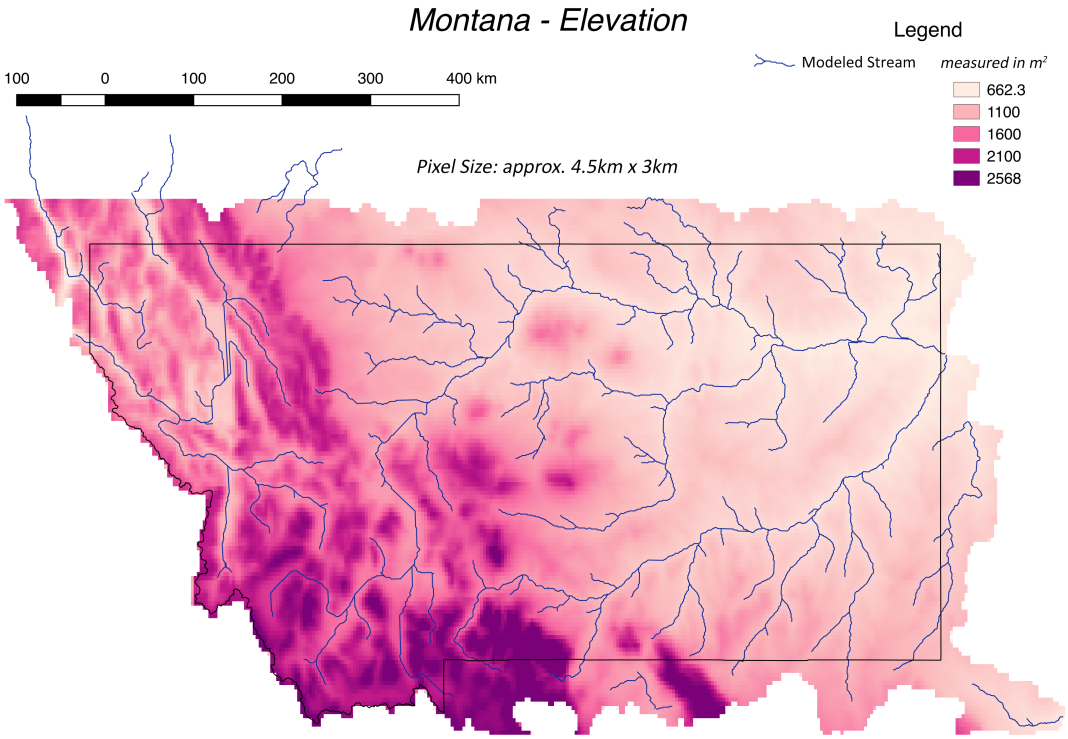


Figure 3-1: Elevation throughout Montana

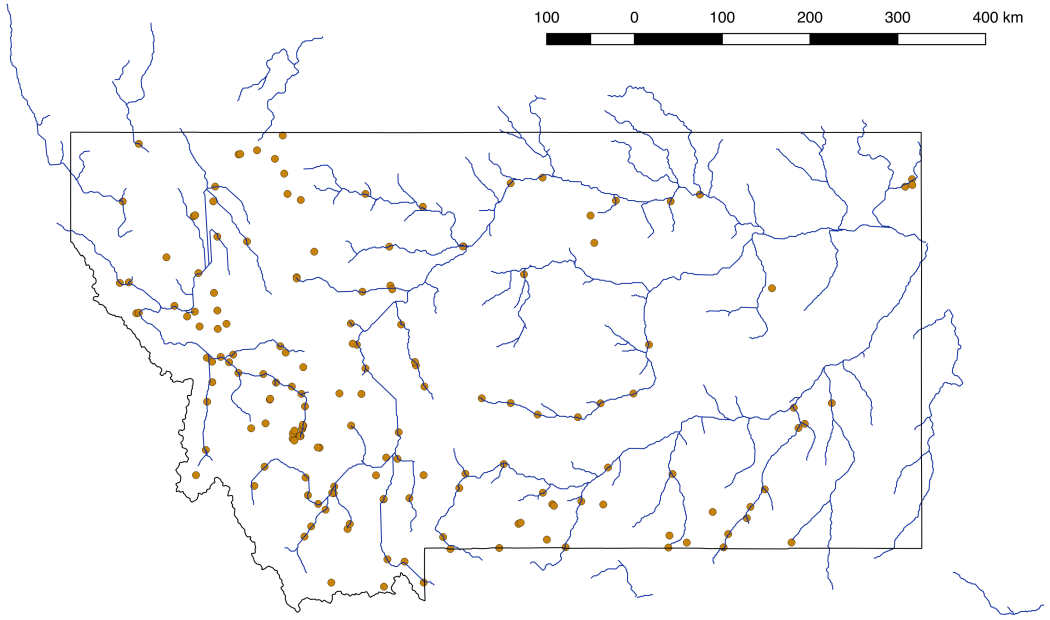


Figure 3-2: all SWE stations plotted against modeled streamflows

# Appendix A

# The Dual Ensemble Kalman Filter

### A.0.1 Prediction Phase

In a Dual Ensemble Kalman filter, each ensemble member $i$ is represented by a stochastic model similar to (2.1). The modified equation is as follows:

$$x_{t+1}^{i-} = f(x_t^{i+}, u_t^i, \theta_t^{i-}) + \omega_t, \quad i = 1, ..., n \tag{A.1}$$

Where $n$ is the total number of ensembles. The $-/+$ superscripts denote corrected $(+)$ and uncorrected $(-)$ values. Note that $\theta_t^{i-}$'s $t$ superscript does not necessarily denote that $\theta$ is time variant but rather indicates that parameter values change as they are filtered over time. The noise term $\omega_t$ accounts for model error and will hereafter be excluded from the state equation.

Errors in the forcing data are accounted for through the perturbation the forcing data vector $u_t$ with random noise $\zeta_t^i$ to generate a unique variable $u_t^i$ for each ensemble. $\zeta_t^i$ is drawn from a normal distribution with a covarience matrix $Q_t^i$.

$$u_{t+1}^i = u_t + \zeta_t^i, \quad \zeta_t^i \sim N(0, Q_t^i) \tag{A.2}$$

To generate the priori parameters $\theta_{t+1}^{i-}$ an evolution of the parameters similar to the evolution of the state variables must be implemented. To accomplish this the kernel smoothing technique developed by West [22] and implemented by Liu [9] is

used. Legacy implementations of parameter evolution added a small perturbation sampled from $N(0, \Sigma_t^\theta)$, where $\Sigma_t^\theta$ represents the covariance matrix of $\theta$ at timestep $t$. This legacy method of evolution resulted in overly disposed parameter samples and the loss of continuity between two consecutive points in time [9] [3]. Kernel smoothing has been used effectively to solve this problem in previous Dual Ensemble Kalman filter implementations [13] and similar models [3].

$$\theta_{t+1}^{i-} = a\theta_t^{i+} + (1-a)\bar{\theta}_t^{+} + \tau_t^i \tag{A.3}$$

$$\tau_t^i = N(0, h^2 V_t) \tag{A.4}$$

Where $\bar{\theta}_t^+$ is the mean of the parameters with respect to the ensembles, $V_t = var(\theta_t^{i+})$, $a$ is a shrinkage factor between (0,1) of the kernel location, and $h$ is a smoothing factor. $h$ is defined by $\sqrt{1-a1/2}$, while $a$ is generally between (.45,.49). Note that $h$ and $a$ tend to vary per model and optimal values for these parameters are generally found via experimentation [13] [1] [2] [3].

### A.0.2 Parameter Correction Phase

In an Ensemble Kalman Filter, observations are perturbed to reflect model error. Therefore, the variable $z_{t+1}^i$ is defined as follows:

$$z_{t+1}^i = z_{t+1} + \eta_{t+1}^i, \quad \eta_{t+1}^i = N(0, R_{t+1}) \tag{A.5}$$

Where $z_{t+1}$ is an observation vector defined by (2.2) and $\eta_{t+1}^i$ is a random perturbation drawn from a normal distribution with covarience matrix $R_{t+1}$. A set of state predictions that can be related to the observations are generated by running the priori state vector through the function $h(.)$:

$$\hat{y}_{t+1}^i = h(x_{t+1}^{i-}, \theta_{t+1}^{i-}) \tag{A.6}$$

The parameter update equation is similar to the update equation of the linear Kalman filter $\hat{x}_t^+ = \hat{x}_t^- + K_t(z_t - H\hat{x}_t)$. Notably, parameters are corrected in lieu of

the states:

$$\theta_{t+1}^{i+} = \theta_{t+1}^{i-} + K_{t+1}^{\theta}(z_{t+1}^{i} - \hat{y}_{t+1}^{i}) \tag{A.7}$$

To facilitate this, $K_{t+1}^{\theta}$ is defined as

$$K_{t+1}^{\theta} = \frac{\Sigma_{t+1}^{\theta,\hat{y}}}{\Sigma_{t+1}^{\hat{y},\hat{y}} + R_{t+1}} \tag{A.8}$$

where $\Sigma_{t+1}^{\theta,\hat{y}}$ is the cross covariance of $\theta_{t+1}$ and $\hat{y}_{t+1}$, $\Sigma_{t+1}^{\hat{y},\hat{y}}$ is the covarience of $\hat{y}_{t+1}$, and $R_{t+1}$ is the observation error matrix from (A.5).

### A.0.3 State Correction Phase

After $\theta_{t+1}^{i+}$ has been calculated the model is run again (A.1) with the $\theta_{t+1}^{i+}$ replacing $\theta_{t+1}^{i-}$.

$$x_{t+1}^{i-} = f(x_{t}^{i+}, u_{t}^{i}, \theta_{t}^{i+}), \quad i = 1, ..., n \tag{A.9}$$

After a new state vector is generated it is re-run through (A.6) with the new parameter vector:

$$\hat{y}_{t+1}^{i} = h(x_{t+1}^{i-}, \theta_{t+1}^{i+}) \tag{A.10}$$

The corrected state vector is then run through the state update equation

$$x_{t+1}^{i+} = x_{t+1}^{i-} + K_{t+1}^{x}(z_{t+1}^{i} - \hat{y}_{t+1}^{i}) \tag{A.11}$$

$$K_{t+1}^{x} = \frac{\Sigma_{t+1}^{x,\hat{y}}}{\Sigma_{t+1}^{\hat{y},\hat{y}} + R_{t+1}} \tag{A.12}$$

where $\Sigma_{t+1}^{x,\hat{y}}$ is the cross covariance of $x_{t+1}$ and $\hat{y}_{t+1}$.

# Appendix B

# daWUAPhydroengine

REWRITE THIS

The hydrologic system is simulated using a rainfall-runoff model coupled to a routing component that simulates streamflows in the regional stream network. We adapted the HBV model [**?**, **?**] to simulate subcatchment-scale hydrologic processes (snowmelt, evapotranspiration, infiltration) and to transform precipitation into runoff and streamflow. Runoff that reaches the channel is routed through the stream network using the Muskingum-Cunge routing algorithm [**?**]. In this appendix we provide here a description of the implementation of the algorithms.

### B.0.1 Rainfall Runoff component

The HVB model [**?**, **?**] is implemented as a mixture of gridded and vector-based operations to leverage the distributed nature of raster meteorological datasets while simultaneously taking advantage of the reduced computational burden of operating over polygons that aggregate runoff production over uniform hydrologic response units (HRUs).

Snowpack accumulation and melt and soil processes are calculated over the uniform raster grid imposed by the meteorological inputs (precipitation, air temperature, and potential evapotranspiration). In the next two paragraphs subscript $i$ indicates that the variable or parameter is spatially distributed and is represented at grid point

$i$. Superscript $t$ indicates that the variable is dynamic and its value is represented at time step $t$. Variables with no script or superscript indicate that the variable is spatially constant or time invariant.

**Precipitation and snowpack processes** Precipitation is partitioned between snowfall and rainfall using minimum and maximum daily air temperatures and a critical temperature threshold $Tc$ that determines the the snow-rain transition:

$$Snow_i^t = \begin{cases} P_i^t & Tmax_i^t < Tc_i \\ P_i^t * \frac{Tc_i - Tmin_i^t}{Tmax_i^t - Tmin_i^t} & Tmin_i^t < Tc_i < Tmax_i^t \\ 0 & Tmin_i^t > Tc_i \end{cases} \tag{B.1}$$

$$Rain_i^t = P_i^t - Snow_i^t \tag{B.2}$$

where $P$ is precipitation ($\mathrm{mm\,d^{-1}}$), $T_{max}$ and $T_{min}$ are maximum and minimum air temperature (°C), $Rain$ is liquid precipitation and $Snow$ is snowfall at pixel $i$ during time step $t$ ($\mathrm{mm\,d^{-1}}$). Snowfall during day $t$ contributes to the snow water equivalent ($SWE$, (mm)) of the snowpack:

$$SWE_i^t = SWE_i^{t-1} + Snow_i^t \Delta t \tag{B.3}$$

The snowpack melt process is simulated using a degree day factor model occurs when average air temperature exceeds a air temperature threshold ($Tm$):

$$Melt_i^t = ddf_i * (Tav_i^t - Tm_i)] \text{ for } Tav_i^t > Tm_i \tag{B.4}$$

$$Rain_i^t = P_i^t - Snow_i^t \tag{B.5}$$

where $Melt$ is the amount of water output from the snowpack ($\mathrm{mm\,d^{-1}}$), $Tav$ is average air temperature over the time step (°C), and $ddf$ is the degree day factor ($\mathrm{mm\,d^{-1}\,°C^{-1}}$), an empirical parameter that represents the snowmelt rate per degree

of air temperature above $Tm$. Any melt form the snowpack during time $t$ is subtracted from the snowpack storage ($SWE$) and added to the amount of water ponded in the surface:

$$Pond_i^t = Pond_i^{t-1} + (Melt_i^t + Rain_i^t)\Delta t \tag{B.6}$$

$$SWE_i^t = SWE_i^t - Melt_i^t \Delta t \tag{B.7}$$

where $Pond$ (mm) is liquid water available on the surface to infiltrate or produce runoff.

**Soil processes** Recharge into the soil system occurs when liquid water ponding the surface infiltrates into the soil. Ponded water that is not infiltrated increases the topsoil compartment that generates fast runoff. The fraction of ponded water that infiltrates into the soil is a exponential function of the relative water storage in the soil:

$$\Delta SM_i^t = Pond_i^t * \left(1 - \frac{SM_i^t}{FC_i^t}\right)^{\beta} \tag{B.8}$$

$$\tag{B.9}$$

where $SM$ (mm) is the amount of water in the soil compartment, $FC$ (mm) is the maximum amount of water soil can hold before water starts percolating to the groundwater system, and *beta* (dimensionless) is an empirical parameter. Simultaneously, actual evapotranspiration ($AET$, $\mathrm{mm\,d^{-1}}$) reduces the amount of water storage in the soil and is also controlled by the degree of saturation of the soil (ration of $SM$ to $FC$).

$$AET_i^t = PET_i^t * \left( \frac{SM_i^t}{FC_i * LP_i} \right)^l \tag{B.10}$$

$$\tag{B.11}$$

where $PET$ is potential evapotranspiration ($\mathrm{mm\,d^{-1}}$)) and $l$ is an empirical dimensionless parameter. Infiltration and actual evapotranspiration control the dynamics of water storage in the soil and amount of surface water that generates fast runoff:

$$SM_i^t = SM_i^t + \Delta SM_i^t - AET_i^t \Delta t \tag{B.12}$$

$$OVL_i^t = Pond_i^t - \Delta SM_i^t \tag{B.13}$$

where $OVL$ (mm) is water that recharges the upper (near-surface) runoff-generating compartment.

**Percolation and runoff generation** Excess water in the topsoil and in two groundwater compartments generate outflow that represent fast and intermediate runoff and baseflow. These processes are implemented at the HRU level. For this, calculations about overland flow generation and soil moisture performed at the grid level are averaged over subwatersheds representing HRUs. Spatial arithmetic averaging soil water storage over all grid cells $i$ contained within a given HRU $j$ is represented using angle brackets $< . >$. The mass balance and percolation of water from the soil upper to the soil lower zone is implemented as:

$$Rech_j^t =< OVL_i^t >_j + < max(SM_i^t - FC_i, 0) >_j \tag{B.14}$$

$$SUZ_j^t = SUZ_j^{t-1} + Rech_j^t + Pond_j^t - Q0_j^t \Delta t - Q1_j \Delta t - PERC_j \tag{B.15}$$

$$SLZ_j^t = SLZ_j^{t-1} + PERC_j - Q2\Delta t \tag{B.16}$$

*Rech* (mm) is water storage in the near-surface compartment that generates fast runoff, $SUZ$ (mm) is the storage in the upper groundwater compartment, and $SLZ$ (mm) is water storage in the lower (deeper) groundwater compartment in HRU $j$ at time step $t$. $Q_0$, $Q_1$, and $Q_2$ ($\mathrm{mm\,d^{-1}}$) are specific runoff rates from the soil surface, and the upper and lower soil zones:

$$Q0_j^t = max((SUZ_j - HL1_j) * \frac{1}{CK0_j}, 0.0) \tag{B.17}$$

$$Q1_j^t = SUZ_j * \frac{1}{CK1_j} \tag{B.18}$$

$$Q2_j^t = SLZ_j * \frac{1}{CK2_j} \tag{B.19}$$

$$Qall_j^t = Q0_j^t + Q1_j^t + Q2_j^t \tag{B.20}$$

where $HL1$ (mm) is an empirical water storage threshold the triggers the generation of fast runoff, and $CK0$, $C10$, $CK2$ (d) are empirical parameters representing the characteristic drainage time of each of the compartments. Total outflow from HRU $j$ on day $t$ is distributed over time to produce the catchment response by convoluting the output of HRU $j$ by triangular standard unit hydrograph with base $M_{base}$.

$$Q_j^t = \sum_{i=1}^{M_{base}} Qall_j^{t-i+1} U(i) \tag{B.21}$$

$$U(i) = \begin{cases} \frac{4}{M_{base}^2} * i & 0 < i < M_{base}/2 \\ -\frac{4}{M_{base}^2} * i + \frac{4}{M_{base}} & M_{base}/2 < i < M_{base} \end{cases} \tag{B.22}$$

where $U$ is a triangular hydrograph of area 1 and a base $MAXBAS$ (d)representing the hydrograph duration .

### B.0.2 Routing component

The response at the end of each $HRU$ is routed through the stream network using the Muskingum-Cunge routing model. In this model the storage in each stream reach $k$ is given by the following discharge-storage equation:

$$S_k^t = K\left[eQ_{in} + (1-e)Q_{out}\right], \tag{B.23}$$

which has parameters $K$ (d) and $e$ (dimensionless) controlling, respectively, the celerity and dispersion of the wave routed through the channel.

Substituting this relationship in a finite-difference form of the continuity equation $\frac{S_j^{t+1}-S_j^t}{\Delta t} = Q_{in} - Q_{out}$ for a multi-reach system with lateral inflows injected upstream of reach draining $HRU$ $j$ at average constant rate through time step $t$ $q_j^{t+1}$ yields:

$$Q_j^{t+1}\left[K_j(1-e_j) + 0.5\Delta t\right] + Q_{j-1}^{t+1}\left[K_j e_j - 0.5\Delta t\right] \tag{B.24}$$

$$= Q_j^t\left[K_j(1-e_j) - 0.5\Delta t\right] + Q_{j-1}^t\left[K_j e_j + 0.5\Delta t\right] \tag{B.25}$$

$$+ q_j^{t+1}\left[K_j(1-e_j) + 0.5\Delta t\right] \tag{B.26}$$

Each of the $HRUs$ contains one reach with an upstream and a downstream node. Streamflows for each of the $j = 1, ..., J$ reaches are integrated over time using a first-order explicit finite difference scheme. The system of $J$ equations can be assembled as a linear system of the form:

$$\mathbf{A}\mathbf{Q}^{\mathbf{t+1}} = \mathbf{B} \tag{B.27}$$

where $\mathbf{Q}^{\mathbf{t+1}}$ is the vector of unknown streamflows at time $t+1$ for each of the $J$ reaches of the network that is solved each time step. Matrices $\mathbf{A}$ add $\mathbf{B}$ are functions

of the model parameters and streamflows at timestep $t$:

$$\mathbf{A} \equiv (\mathbf{a} + \mathbf{\Phi b})^T \tag{B.28}$$

$$\mathbf{B} \equiv (\mathbf{d} + \mathbf{\Phi c})^T \mathbf{Q}^t + \mathbf{I}(\mathbf{a} \odot \mathbf{q}^{t+1}) \tag{B.29}$$

where $\mathbf{\Phi}$ is a $JxJ$ sparse connectivity (0,1)-matrix where the elements indicate if two pairs of nodes are connected. Flow direction is from nodes in the rows to nodes in the columns. Rows representing the upstream node of $HRUs$ that drain an outlet node (exit the domain) are all zero. Finally,

$$\mathbf{a} = \mathbf{I}(\mathbf{K} - \mathbf{K} \odot \mathbf{e}) + dt * 0.5 \tag{B.30}$$

$$\mathbf{b} = \mathbf{I}(\mathbf{K} \odot \mathbf{e}) - dt * 0.5 \tag{B.31}$$

$$\mathbf{c} = \mathbf{I}(\mathbf{K} - \mathbf{K} \odot \mathbf{e}) - dt * 0.5) \tag{B.32}$$

$$\mathbf{d} = \mathbf{I}(\mathbf{K} \odot \mathbf{e}) + dt * 0.5 \tag{B.33}$$

where $\mathbf{K}$ is the identity matrix of order $J$, $\mathbf{K}$ and $\mathbf{e}$ are column vectors holding parameters $K$ and $e$ for each of the $N$ reaches in the network. The $\odot$ operator denotes the Schur (elementwise) product between two vectors. The solution of (B.27) becomes unstable if $\Delta t > 2 * K_j * (1 - e_j)$. To ensure robust and stable solution an adaptive time stepping scheme was implemented. In this scheme, the default time step is reduced by an integer fraction until the the stability condition is satisfied in all reaches.

# Bibliography


[1] Jeffrey L. Anderson, Stephen L. Anderson, Jeffrey L. Anderson, and Stephen L. Anderson. A Monte Carlo Implementation of the Nonlinear Filtering Problem to Produce Ensemble Assimilations and Forecasts. *Monthly Weather Review*, 127(12):2741–2758, dec 1999.

[2] J. D. Annan, D. J. Lunt, J. C. Hargreaves, and P. J. Valdes. Parameter estimation in an atmospheric GCM using the Ensemble Kalman Filter. *Nonlinear Processes in Geophysics*, 12(3):363–371, feb 2005.

[3] M. Chen, S. Liu, L.L. Tieszen, and D.Y. Hollinger. An improved state-parameter analysis of ecosystem models using data assimilation. *Ecological Modelling*, 219(3-4):317–326, dec 2008.

[4] Geir Evensen. Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research*, 99(C5):10143, may 1994.

[5] H. J. Hendricks Franssen and W. Kinzelbach. Real-time groundwater flow modeling with the Ensemble Kalman Filter: Joint estimation of states and parameters and the filter inbreeding problem. *Water Resources Research*, 44(9), sep 2008.

[6] Andrew H. Jazwinski. Stochastic Processes and Filtering. *Mathematics in Science and Enginerring*, 1970.

[7] Simon J. Julier and Jeffrey K. Uhlmann. New extension of the Kalman filter to nonlinear systems. volume 3068, page 182. International Society for Optics and Photonics, jul 1997.

[8] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35, mar 1960.

[9] Fang Liu and Fang Liu. Bayesian Time Series: Analysis Methods Using Simulation-Based Computation. 2000.

[10] Yuqiong Liu and Hoshin V. Gupta. Uncertainty in hydrologic modeling: Toward an integrated data assimilation framework. *Water Resources Research*, 43(7), jul 2007.

[11] Stefano Mariani and Alberto Corigliano. Impact induced composite delamination: state and parameter identification via joint and dual extended Kalman filters. *Computer Methods in Applied Mechanics and Engineering*, 194(50-52):5242–5272, dec 2005.

[12] Robert N. Miller, Michael Ghil, and François Gauthiez. Advanced Data Assimilation in Strongly Nonlinear Dynamical Systems. *Journal of the Atmospheric Sciences*, 51(8):1037–1056, apr 1994.

[13] Hamid Moradkhani, Soroosh Sorooshian, Hoshin V. Gupta, and Paul R. Houser. Dual state-parameter estimation of hydrological models using ensemble Kalman filter. *Advances in Water Resources*, 28(2):135–147, feb 2005.

[14] Jason W. Osborne. Advantages of hierarchical linear modeling. Osborne, Jason W. *Research & Evaluation*, 7(1), 2000.

[15] Rolf H. Reichle. Data assimilation methods in the Earth sciences. *Advances in Water Resources*, 31(11):1411–1418, nov 2008.

[16] Soroosh Sorooshian and John A. Dracup. Stochastic parameter estimation procedures for hydrologie rainfall-runoff models: Correlated and heteroscedastic error cases. *Water Resources Research*, 16(2):430–442, apr 1980.

[17] Soroosh Sorooshian, Qingyun Duan, and Vijai Kumar Gupta. Calibration of rainfallâĂŘrunoff models: Application of global optimization to the Sacramento Soil Moisture Accounting Model. *Water Resources Research*, 29(4):1185–1194, apr 1993.

[18] Peter A. Troch, Claudio Paniconi, and Dennis McLaughlin. Catchment-scale hydrological modeling and data assimilation. *Advances in Water Resources*, 26(2):131–135, feb 2003.

[19] Jasper A. Vrugt, Cees G. H. Diks, Hoshin V. Gupta, Willem Bouten, and Jacobus M. Verstraten. Improved treatment of uncertainty in hydrologic modeling: Combining the strengths of global optimization and data assimilation. *Water Resources Research*, 41(1), jan 2005.

[20] Thorsten Wagener and Howard S. Wheater. Parameter estimation and regionalization for continuous rainfall-runoff models including uncertainty. *Journal of Hydrology*, 320(1-2):132–154, mar 2006.

[21] Xian-Huan Wen and Wen H. Chen. Real-Time Reservoir Model Updating Using Ensemble Kalman Filter With Confirming Option. *SPE Journal*, 11(04):431–442, dec 2006.

[22] Mike West. Mixture models, Monte Carlo, Bayesian updating, and dynamic models. *Computing Science and Statistics*, 1993.