William Cook

Machine Learning

Final Project

Due 12/14/2018

0.1 Abstract

The **daWUAP** team at the University of Montana has created a hydrological rainfall-runoff model that predicts streamflows across the state of Montana. **daWUAPhydroengine** is informed by a variety of smaller models, including a groundwater model, snow water equivalient (swe) model, and agricultural component model. This paper focuses on the calibration of **daWUAPhydroengine** via a Dual State Parameter Estimation Ensemble Kalman Filter. Uniquely, this filter operates on multiple high dimensional parameters represented as raster images. These raster images are both ingested and outputted by the **daWUAPhydroengine** and were previously set at a constant value. The filter aims to both calibrate and distribute these variables geospatially.

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. Many observations such as SWE, streamflow, and precipitation are collected on a daily basis across various geographic regions, allowing the previous day's 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.

Models that ingest data sequentially can be 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[7] was created to solve linear problems and more complicated implementations must be used to solve non-linear problems. The extended Kalman Filter[5] works for mildly non-linear systems but does not function optimally on heavily non-linear systems[9]. The Unscented Kalman Filter[6] 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.

In order to calibrate parameters within a hydrologic model a Dual State Kalman Filter may be used as demonstrated by Moradkhani et. al in 2005 [10]. Dual state Kalman filters add a small perturbation to a series of parameters that the userwishes to calibrate. These perturbed parameters vectors are then corrected in a similar fashion to the state vectors. After this happens a 'second' filter runs and corrects the state vectors in the normal fashion. The Dual State Ensemble Kalman Filter implemented by Moradkhani et. al[10] extends the Ensemble Kalman Filter into a dual state configuration and successfully predicts a set of parameters.

In this paper hierarchical modeling techniques are combined with a 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 spatially similar. To examine the Hierarchical Dual State Ensemble Kalman Filter's application to high dimensional geospatially distributed raster data Professor Marko Maneta's **daWUAPhydroengine**, a hydrological rainfall-runoff model that predicts streamflows across the state of Montana, is used. **daWUAPhydroengine** is informed by a variety of smaller models, including a groundwater model, snow water equivalient (swe) model, and agricultural component model. Each of these smaller models are influenced by a set of uncalibrated high-dimensional parameters that are stored as 2D raster data. Despite this complexity, **daWUAPhydroengine** is designed to be a quick and efficient model. Accordingly, a Hierarchical Dual State Ensemble Kalman Filter is an optimal calibration algorithm to calibrate this raster data because 1) the HDEnKF does not have to compute the high dimensional state covariance matrix and 2) **daWUAPhydroengine** is a sequential model.

Chapter 2 covers the methods behind the Hierarchical Dual Ensemble Kalman Filtering algorithm. Chapter 3 discusses the application of a Hierarchical Dual Ensemble Kalman Filter to **daWUAPhydroengine**. Chapter 4 discusses results and 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 θ_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 ε_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, θ_t represents the true parameters, $h(.)$ is a function that determines the relationship between observation and state vectors, and δ_t represents observation error. δ_t is Gaussian and independent of ε_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 Appendix A for methods), 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 θ_t^{i-}'s t subscript does not necessarily denote that θ is time variant but rather indicates that parameter values change as they are filtered over time. The noise term ω_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 ζ_t^i to generate a unique variable u_t^i for each ensemble. ζ_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 θ_{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 Σ_t^θ represents the covariance matrix of θ 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 [8] [3]. To overcome this the kernel smoothing technique developed by West [12] and implemented by Liu [8] has been used effectively in previous Dual Ensemble Kalman filter implementations [10] 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 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 [10] [1] [2] [3].

In a Hierarchical Duel Ensemble Kalman Filter, parameter perturbation is accomplished using a hierarchical algorithm. For a simple overview of hiearcical models refer to Osborne [11]. 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 η^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^θ_{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 θ_{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 θ^{i+}_{t+1} has been calculated the model is run again (2.3) with the θ^{i+}_{t+1} replacing θ^{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

Results

3.1 Application of DEnHKF to Hydrologic Model

3.1.1 daWUAPhydroengine

The **daWUAPhydroengine** hydrologic model is designed to be implemented in any geographic location. For this study it was utilized to model streamflows throughout the state of Montana. **daWUAPhydroengine** takes precipitation, minimum temperatures, and maximum temperatures as inputs and outputs streamflow data along with some additional states such as snow water equivalent. Since Montana is such a large geographical area sections of the landscape differ in various ways (soil composition, forestation, etc.) To account for this, a series of geospatially distributed raster parameters are calibrated by the filter. Since each parameter is corrected by model state predictions and observations raster images were separated into distinct theissen raster regions, each with an observation point at its center.

Table 3.1: Calibrated Parameters

Parameter (θ)	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

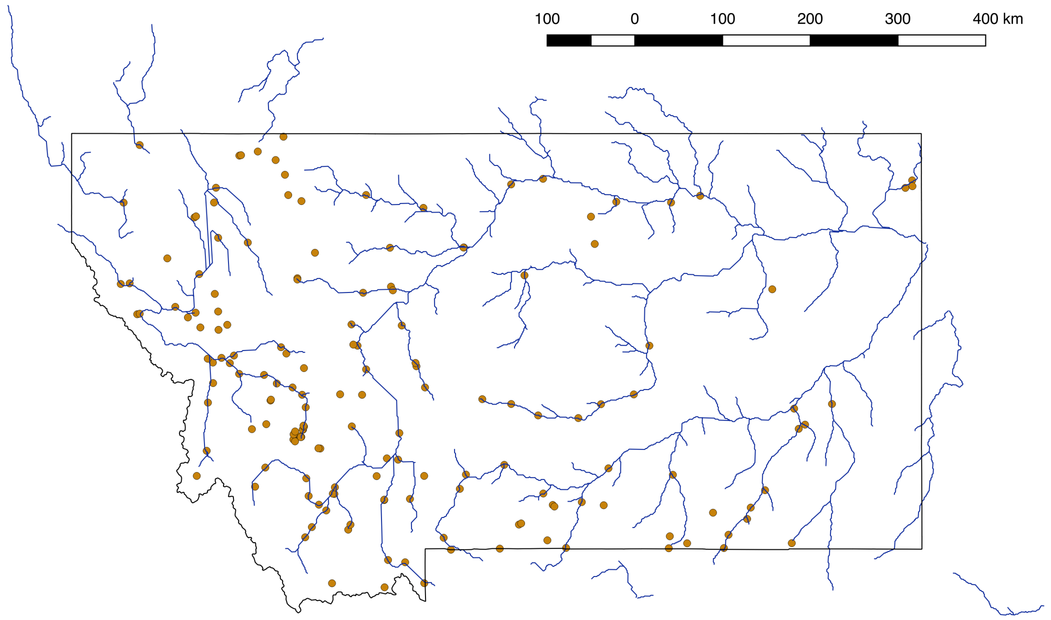

Figure 3-1: all SWE stations plotted against modeled streamflows

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

To apply the filter to **daWUAPhydroengine** minimum temperature, maximum temperature, and precipitation were treated as forcing data.

To implement the Dual Ensemble Kalman Filter with these parameters and forcing data it was necessary to set a minimum and maximum cutoff point for random perturbation (temperatures under 0 Kelvin, for example, break the filter.) Although this method worked well for the forcing data the maximum and minimum cutoffs broke the calibrated parameters. This is because the filter would sometimes correct above or below the parameter values, which resulted in a collapse of the ensemble variance. Since ensemble variance is used to determine the variance of the next timestep's random perturbations the filter could not recover. Two solutions were implemented

Table 3.3: States

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

to keep this from happening: 1) A 'minimum variance' was specified and 2) When one or more values would be corrected past a minimum or maximum cutoff the whole ensemble would simply not correct for that iteration.

Each run involved 100-200 ensembles and took anywhere from 20 hours to 2 days to run. The filter initially did not converge correctly when the kernel smoothing a and h values suggested by Moradkhani et al. were used. When the stronger set of values suggested by Chen et al. [3] were used the parameters converged nicely. Although it was initially worried that stronger kernel smoothing would not allow the ensembles to converge to the correct value values seem constant and consistent with other examples of the literature.

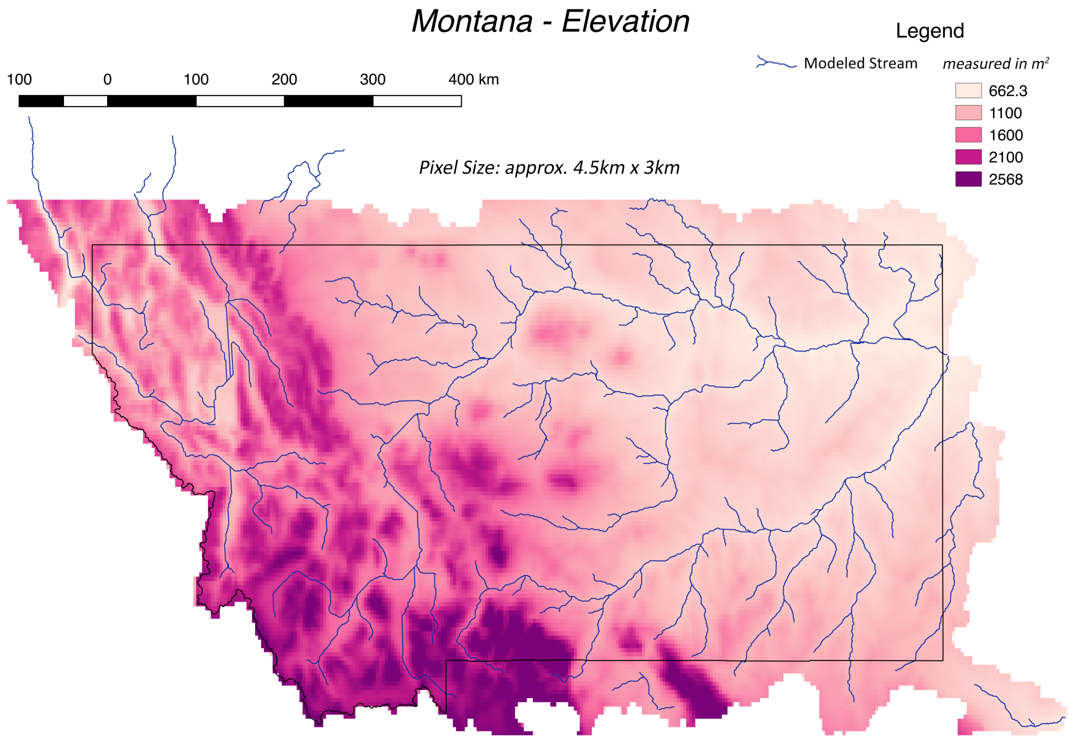

Figure 3-2: Elevation throughout Montana

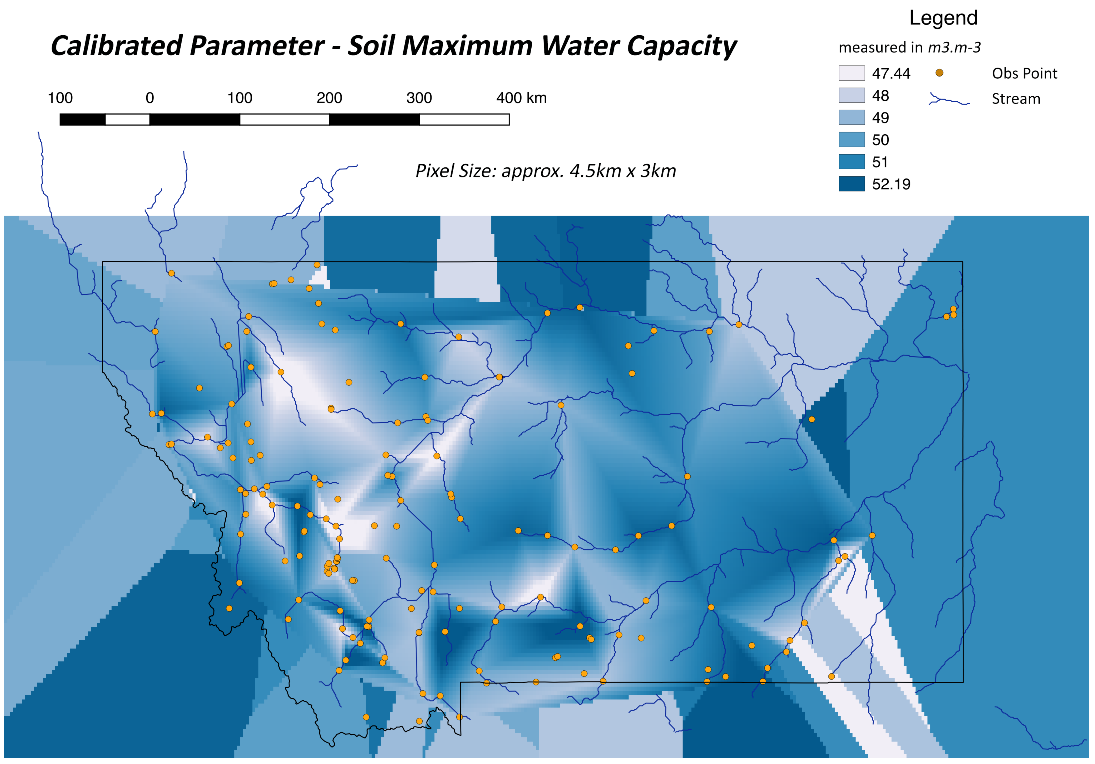
Calibrated Parameter - Soil Maximum Water Capacity
100
0
100
200
300
400 km
Pixel Size: approx. 4.5km x 3km
Legend
measured in m3.m-3
47.44
48
49
50
51
52.19
Obs Point
Stream

Figure 3-3: A calibrated raster - maximum water capacity

3-3 is a visualization of the resulting raster of the calibrated soil max water content parameter. 3-4 is a visualization of the resulting raster raster of the soil beta parameter. When contrasting these with 3-2 it becomes clear that these parameters are not correlated with elevation. If this was the case, it would point to a flaw in **daWUAPhydroengine**'s algorithms. Instead, parameter values are spaced evenly and do not form a trival pattern. This is optimal, as these parameters make up for real world factors that **daWUAPhydroengine**'s methods do not take into account.

3-5 shows the convergence process of three streamflow parameters: AET LP, Soil Beta, and Soil Max Water. The confidence interval is substituted for the ensemble cloud since the amount of perturbation per time step is derived from the variance over all ensembles. Each ensemble converges very nicely.

3-6 display stream observations compared to two instances of **daWUAPhydroengine**, one using the calibrated parameters and one using the default parameters. Both models were started without spun up parameters (e.g both models started with

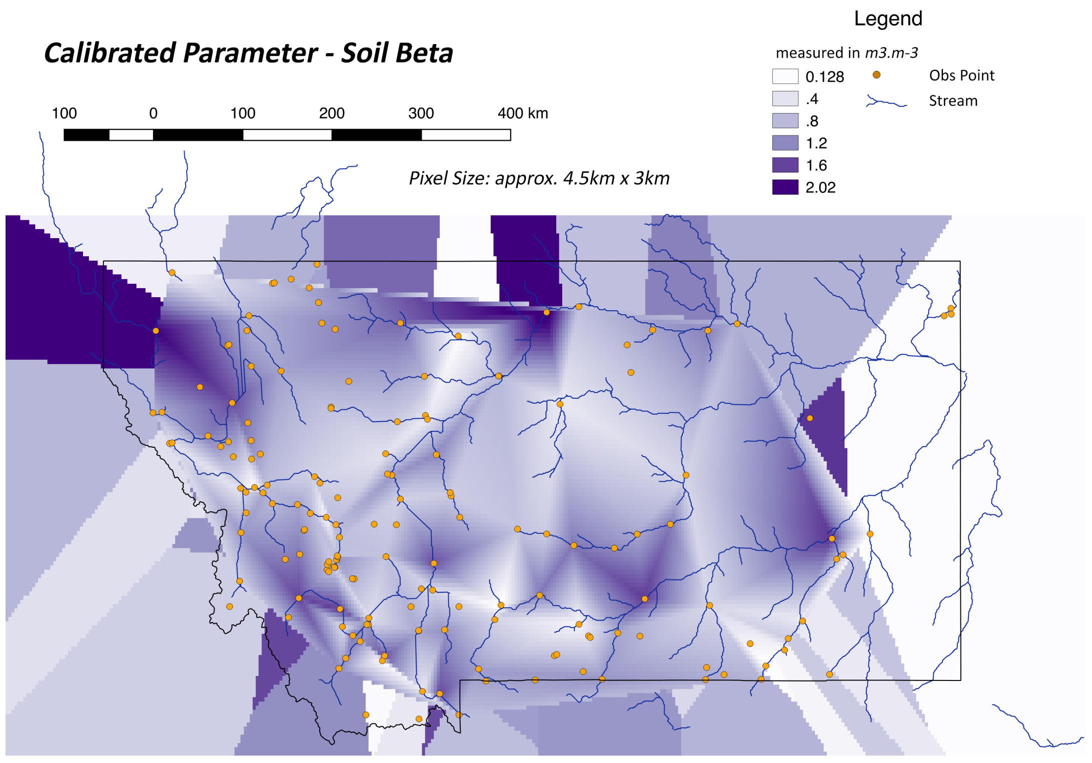

Figure 3-4: A calibrated raster - maximum water capacity

0 for each streamflow and have to converge with the real world.) Note how much more successful the calibrated model behaves.

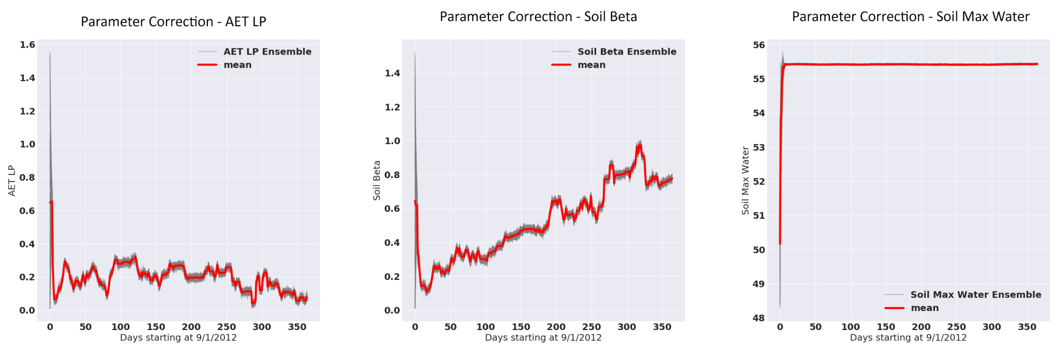

Figure 3-5: Parameter convergences over time

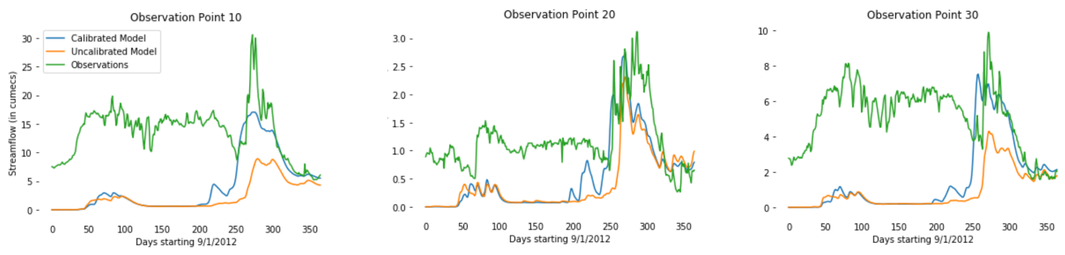

Figure 3-6: A calibrated raster - maximum water capacity

Chapter 4

Discussion

Currently, Prof. Johnson and Prof. Maneta and I are working on a way to combine a Hierarchical regression techinique with this Kalman filter. We believe that this will allow us to switch from the interpolation scheme shown earlier in this paper to a scheme where catchments are used as parameter zones.

4.1 Hierarchical Kalman Filter

The old equation for the evolution of priori parameters is:

$$\theta_{t+1}^{i-} = a\theta_t^{i+} + (1-a)\bar{\theta}_t^{+} + \tau_t^i \tag{4.1}$$

$$\tau_t^i = N(0, h^2 V_t) \tag{4.2}$$

$$V_t = var(\theta_{t+1}) \tag{4.3}$$

Here is a draft of the hierarchical evolution:

$$\theta_{t+1}^{i-} = \alpha_t^{i-} C_t^i + (1-\alpha_t^{i-}) G_t^i + \tau_t^i \tag{4.4}$$

$$C_t^i = a\theta_t^{i+} + (1-a)\bar{\theta}_t^{+} \tag{4.5}$$

$$G_t^i = a\bar{\theta}_t^{i+} + (1-a)\bar{\bar{\theta}}_t^{+} \tag{4.6}$$

$$\tau_t^i = N(0, h^2 V_t) \tag{4.7}$$

$$V_t = \alpha_t^{i-} var(h^2 \theta_{t+1}) + (1 - \alpha_t^{i-}) var(h^2 G_t^i) \tag{4.8}$$

$$\alpha_{t+1}^{i-} = a\alpha_t^{i+} + (1 - a)\bar{\alpha}_t^{+} + \delta_t^i \tag{4.9}$$

$$\delta_t = N(0, h^2{}_\alpha V_t) \tag{4.10}$$

$${}_\alpha V_t = var(\alpha_t) \tag{4.11}$$

Where initial α, m, and n are variables choosable by the user. This is a work in progress.

Chapter 5

Conclusion

In conclusion, this Dual Ensemble Kalman Filter was sucessfully implemented on a hydrologic model. The resulting parameters do lead to better estimation of hydrologic components. Although these results are good, we hope to eventually implement an innovative new Hierarchical Kalman filter that will calibrate parameters at the catchment level.

Appendix A

Appendix A

A.1 The Dual Ensemble Kalman Filter

A.1.1 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 θ_t^{i-}'s t superscript does not necessarily denote that θ is time variant but rather indicates that parameter values change as they are filtered over time. The noise term ω_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 ζ_t^i to generate a unique variable u_t^i for each ensemble. ζ_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 θ_{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 [12] and implemented by Liu [8] is used. Legacy implementations of parameter evolution added a small perturbation sampled from $N(0, \Sigma_t^\theta)$, where Σ_t^θ represents the covariance matrix of θ 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 [8] [3]. Kernel smoothing has been used effectively to solve this problem in previous Dual Ensemble Kalman filter implementations [10] 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 [10] [1] [2] [3].

A.1.2 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 η_{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}^{θ} 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 θ_{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.1.3 State Correction Phase

After θ_{t+1}^{i+} has been calculated the model is run again (A.1) with the θ_{t+1}^{i+} replacing θ_{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 (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 (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 (mm d^{-1}), Tav is average air temperature over the time step (°C), and ddf is the degree day factor (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 (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:

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

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

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

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:

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

where $\mathbf{Q^{t+1}}$ 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] Andrew H. Jazwinski. Stochastic Processes and Filtering. *Mathematics in Science and Enginerring*, 1970.

[6] 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.

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

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

[9] 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.

[10] 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.

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

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