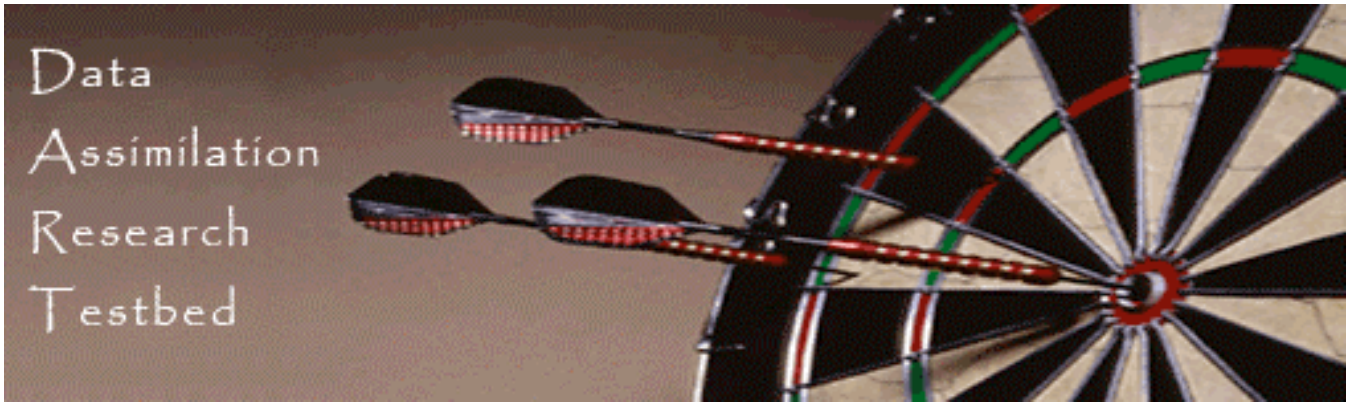
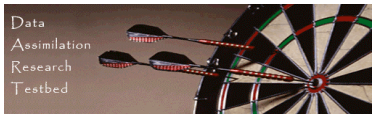
Data Assimilation Research Testbed Tutorial



Section 2: How should observations of a state variable impact an unobserved state variable? Multivariate assimilation.



Single observed variable, single unobserved variable

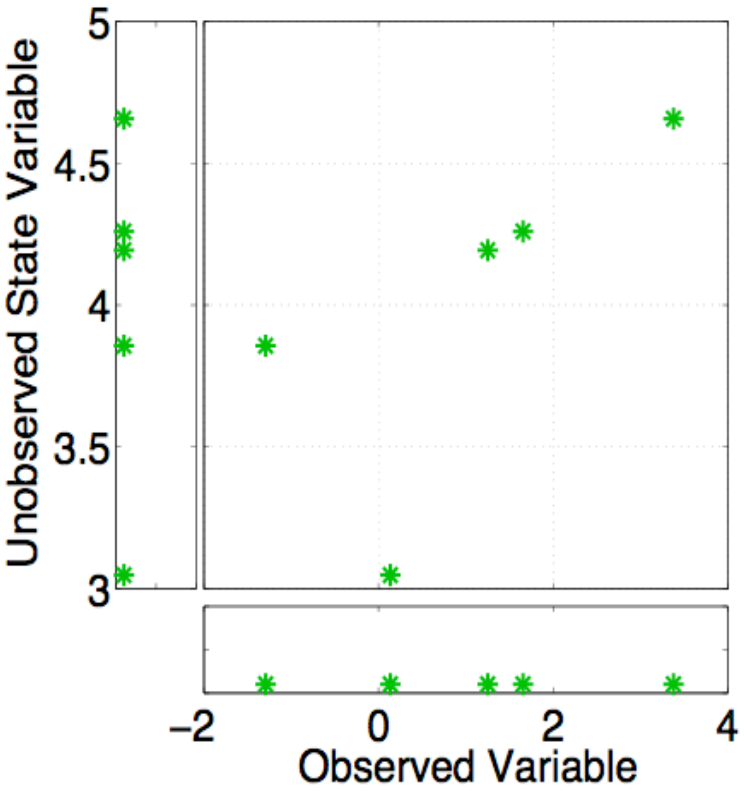
So far, we have a known observation likelihood for single variable.

Now, suppose the prior has an additional variable.

We will examine how ensemble members update the additional variable.

Basic method generalizes to any number of additional variables.

Ensemble filters: Updating additional prior state variables

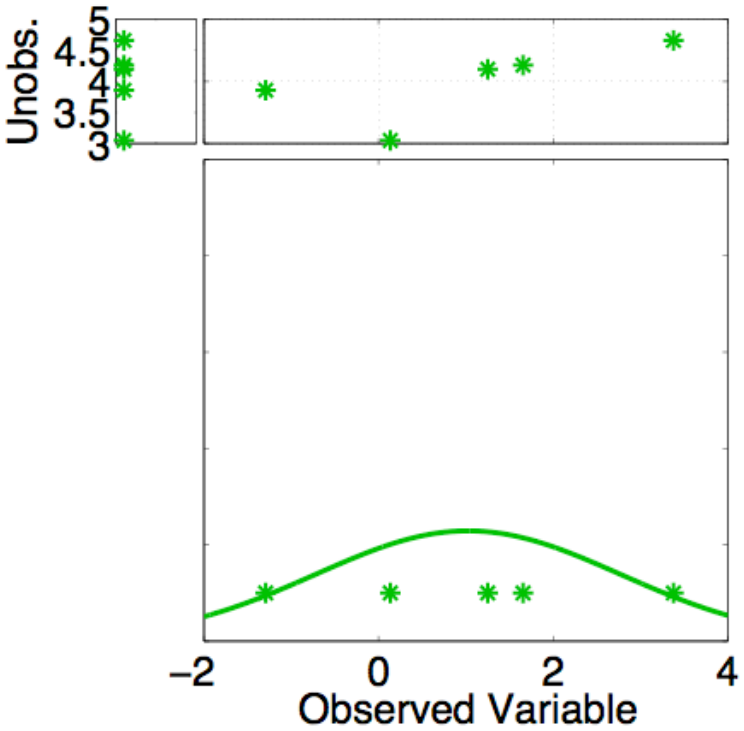


Assume that all we know is prior joint distribution.

One variable is observed.

What should happen to the unobserved variable?

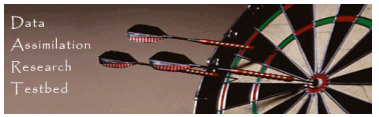
Ensemble filters: Updating additional prior state variables



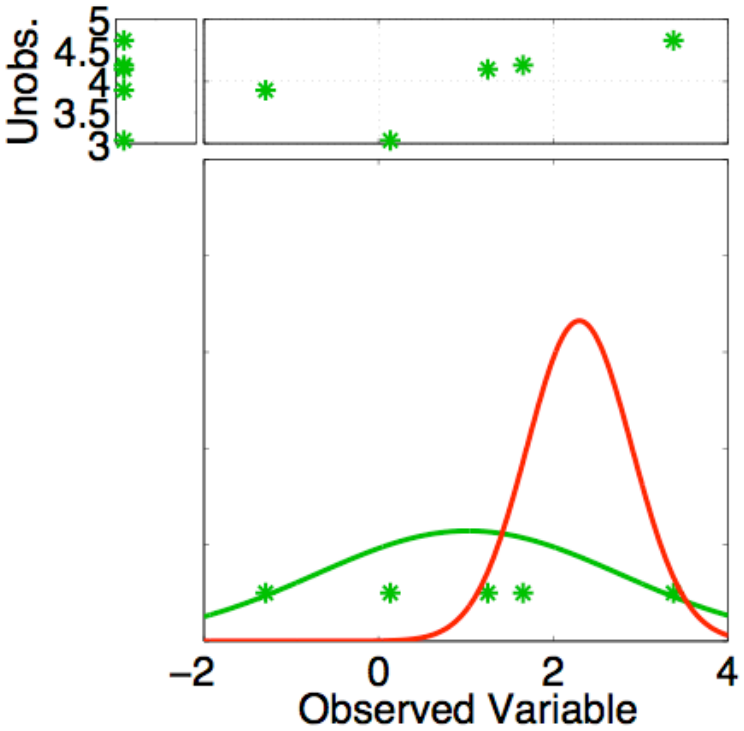
Assume that all we know is prior joint distribution.

One variable is observed.

Update observed variable with one of previous methods.



Ensemble filters: Updating additional prior state variables

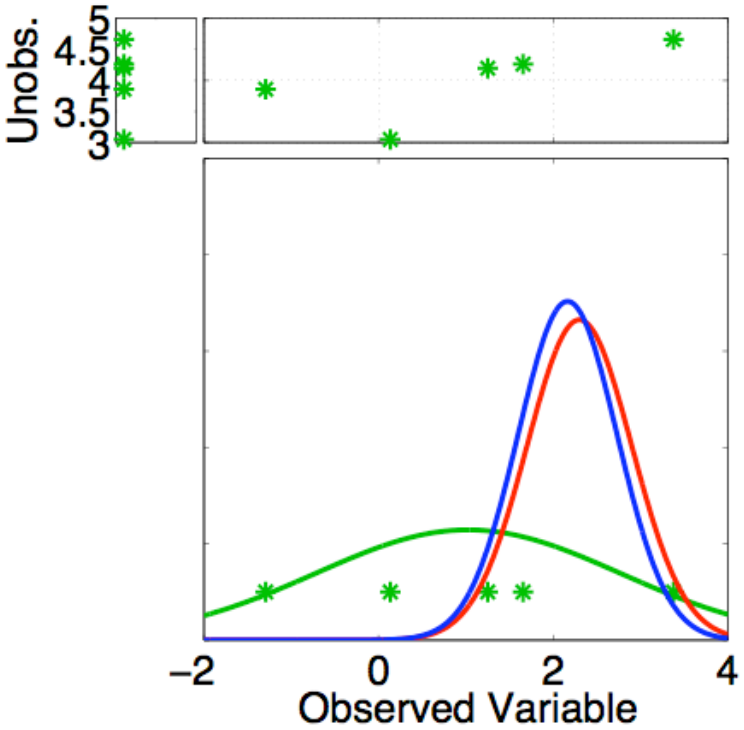


Assume that all we know is prior joint distribution.

One variable is observed.

Update observed variable with one of previous methods.

Ensemble filters: Updating additional prior state variables

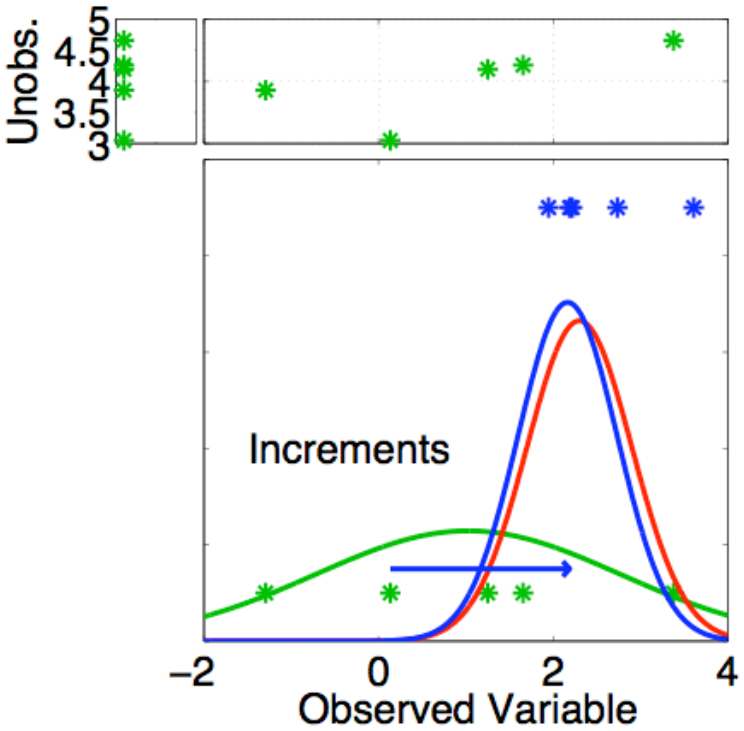


Assume that all we know is prior joint distribution.

One variable is observed.

Update observed variable with one of previous methods.

Ensemble filters: Updating additional prior state variables

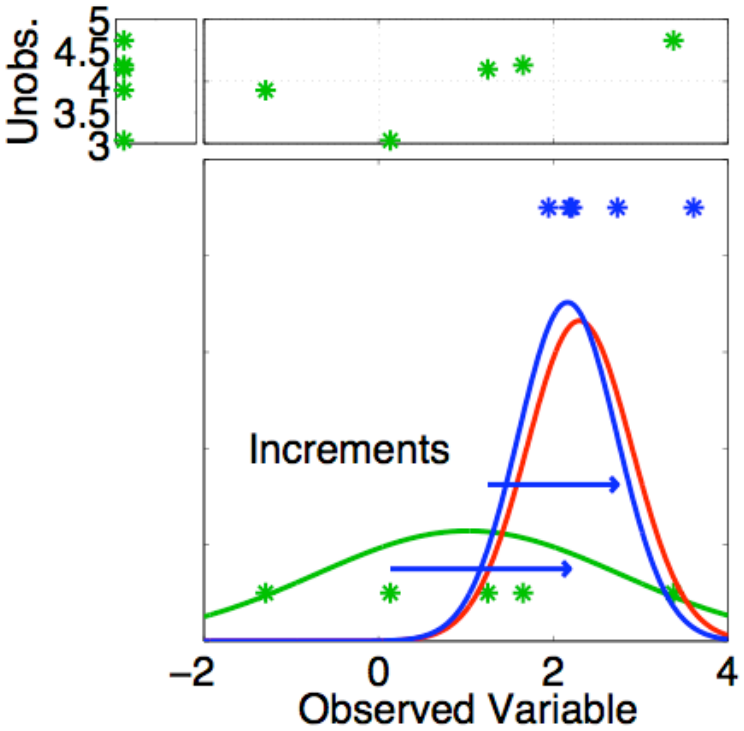


Assume that all we know is prior joint distribution.

One variable is observed.

Compute increments for prior ensemble members of observed variable.

Ensemble filters: Updating additional prior state variables

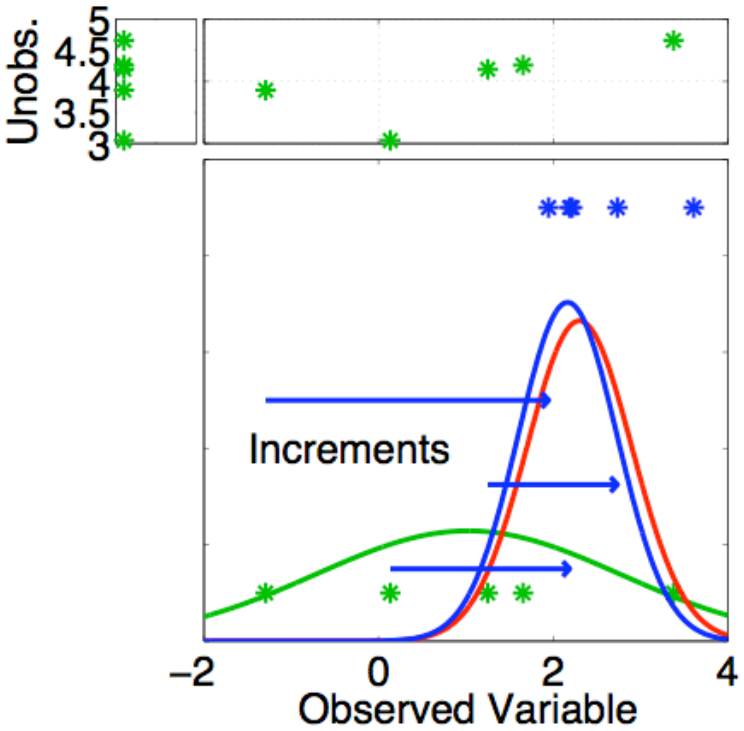


Assume that all we know is prior joint distribution.

One variable is observed.

Compute increments for prior ensemble members of observed variable.

Ensemble filters: Updating additional prior state variables

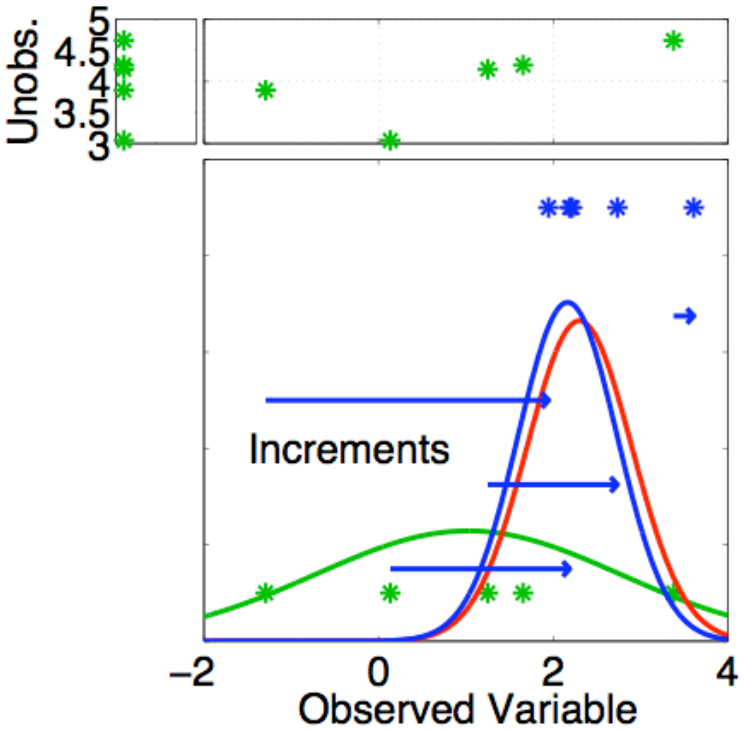


Assume that all we know is prior joint distribution.

One variable is observed.

Compute increments for prior ensemble members of observed variable.

Ensemble filters: Updating additional prior state variables

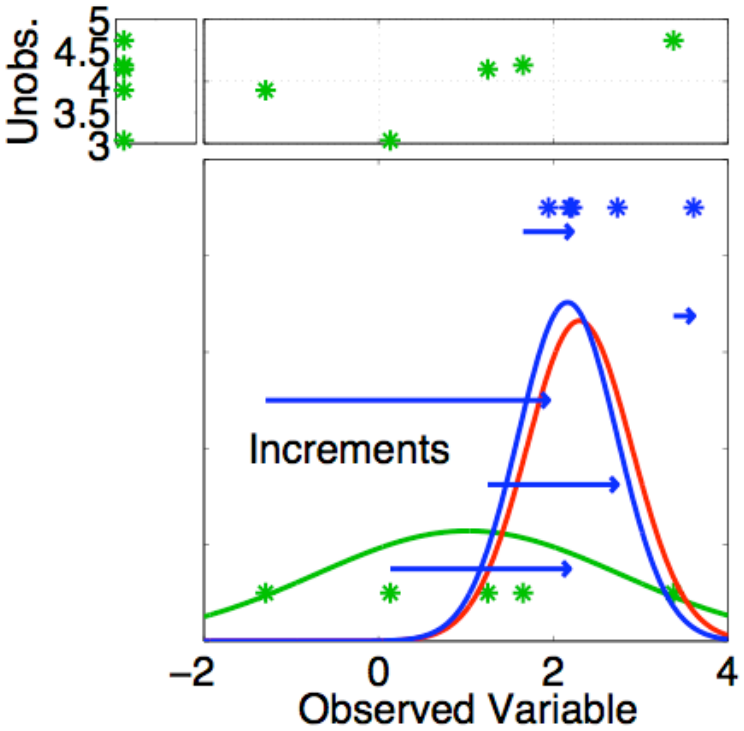


Assume that all we know is prior joint distribution.

One variable is observed.

Compute increments for prior ensemble members of observed variable.

Ensemble filters: Updating additional prior state variables

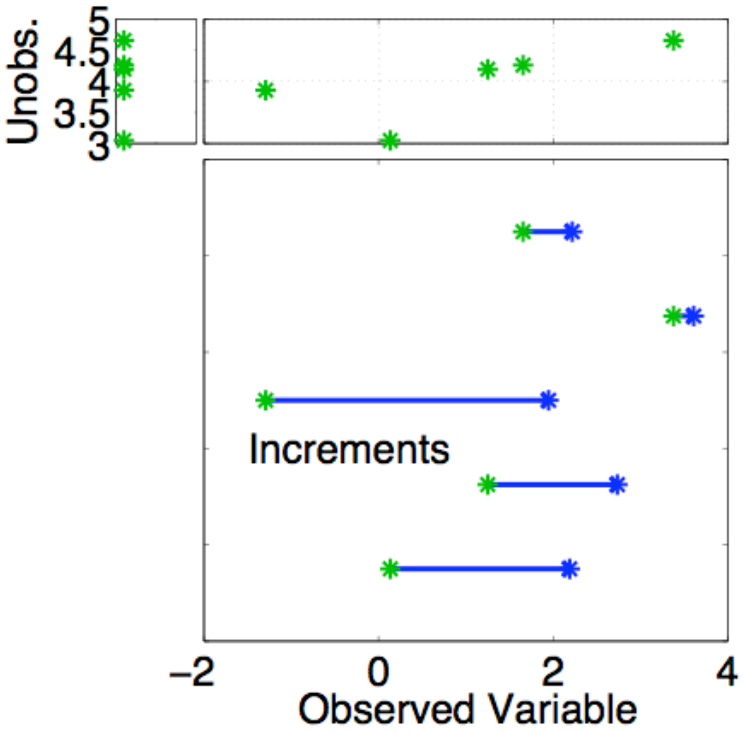


Assume that all we know is prior joint distribution.

One variable is observed.

Compute increments for prior ensemble members of observed variable.

Ensemble filters: Updating additional prior state variables

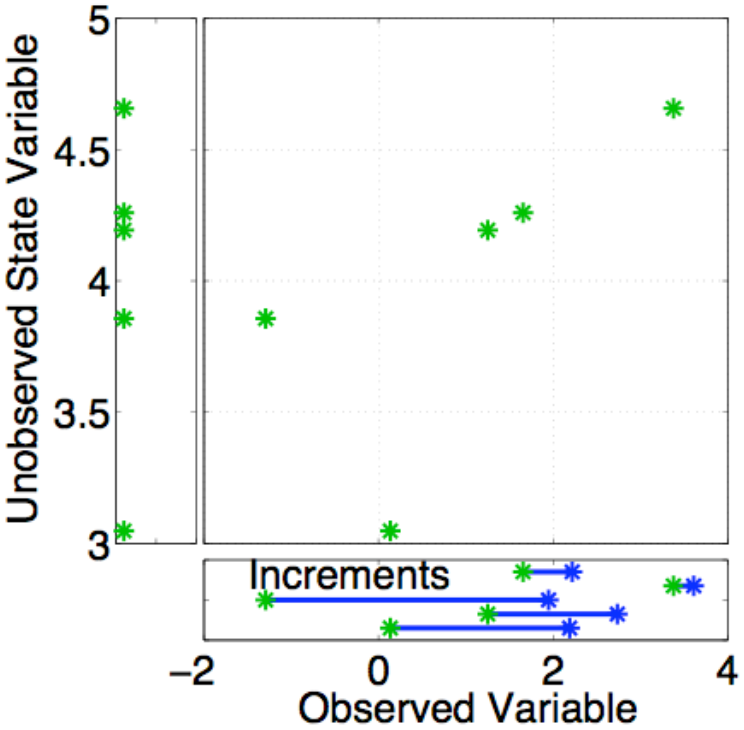


Assume that all we know is prior joint distribution.

One variable is observed.

Using only increments guarantees that if observation had no impact on observed variable, unobserved variable is unchanged (highly desirable).

Ensemble filters: Updating additional prior state variables



Assume that all we know is prior joint distribution.

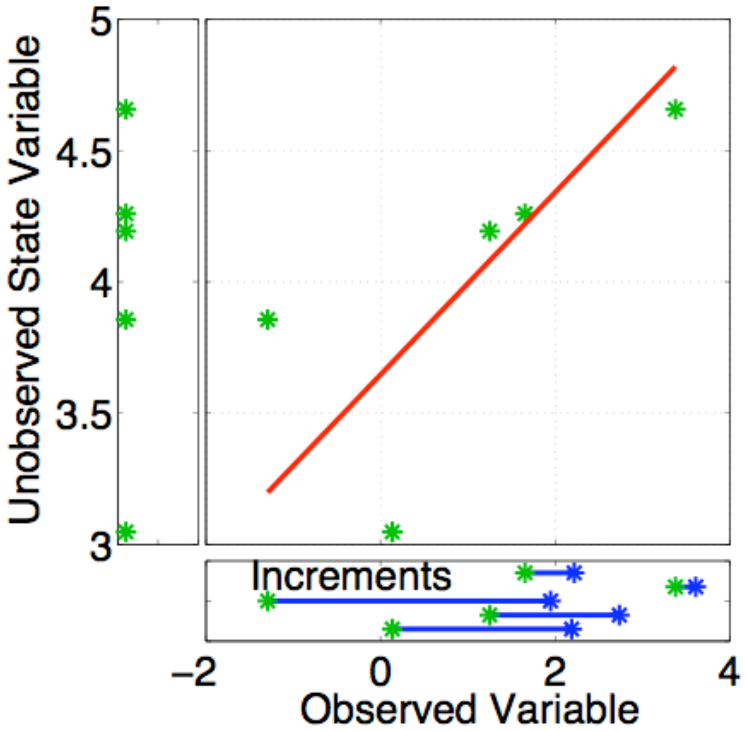
How should the unobserved variable be impacted?

First choice: least squares.

Equivalent to linear regression.

Same as assuming binormal prior.

Ensemble filters: Updating additional prior state variables



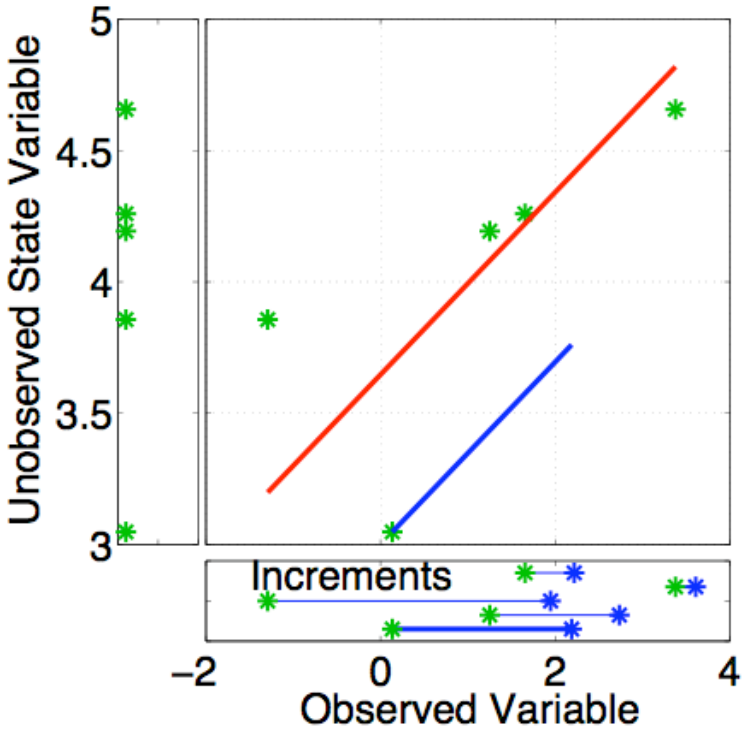
Have joint prior distribution of two variables.

How should the unobserved variable be impacted?

First choice: least squares.

Begin by finding least squares fit.

Ensemble filters: Updating additional prior state variables

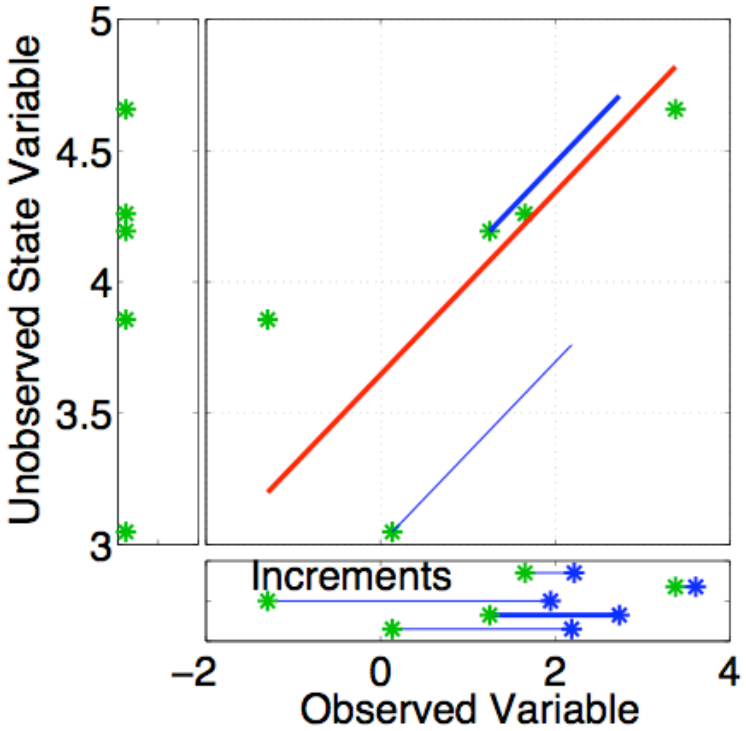


Have joint prior distribution of two variables.

Next, regress the observed variable increments onto increments for the unobserved variable.

Equivalent to first finding image of increment in joint space.

Ensemble filters: Updating additional prior state variables

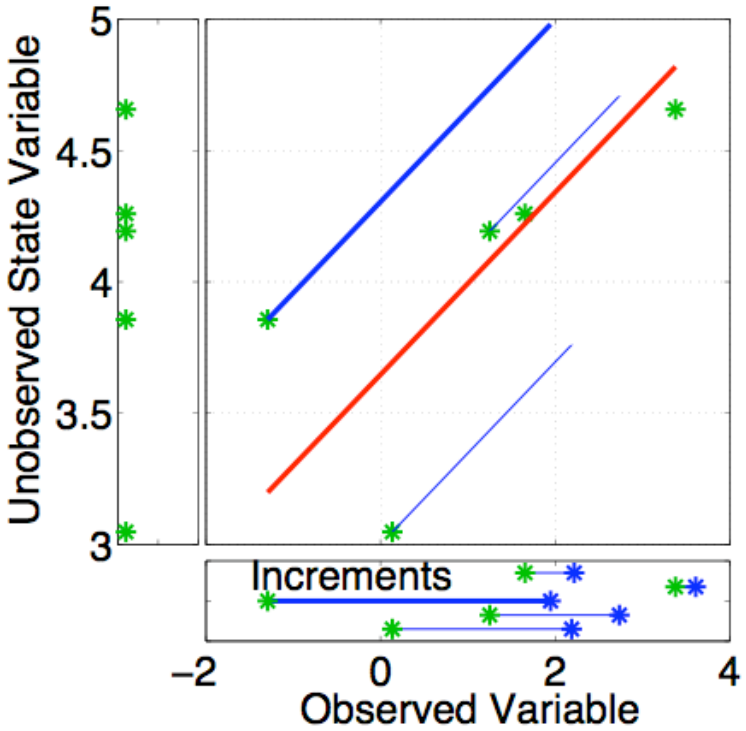


Have joint prior distribution of two variables.

Next, regress the observed variable increments onto increments for the unobserved variable.

Equivalent to first finding image of increment in joint space.

Ensemble filters: Updating additional prior state variables

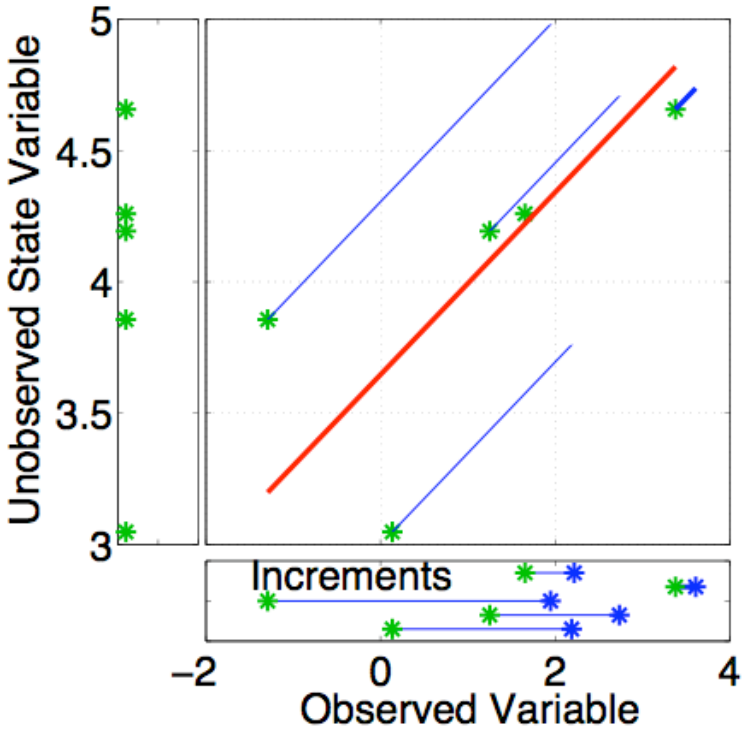


Have joint prior distribution of two variables.

Next, regress the observed variable increments onto increments for the unobserved variable.

Equivalent to first finding image of increment in joint space.

Ensemble filters: Updating additional prior state variables

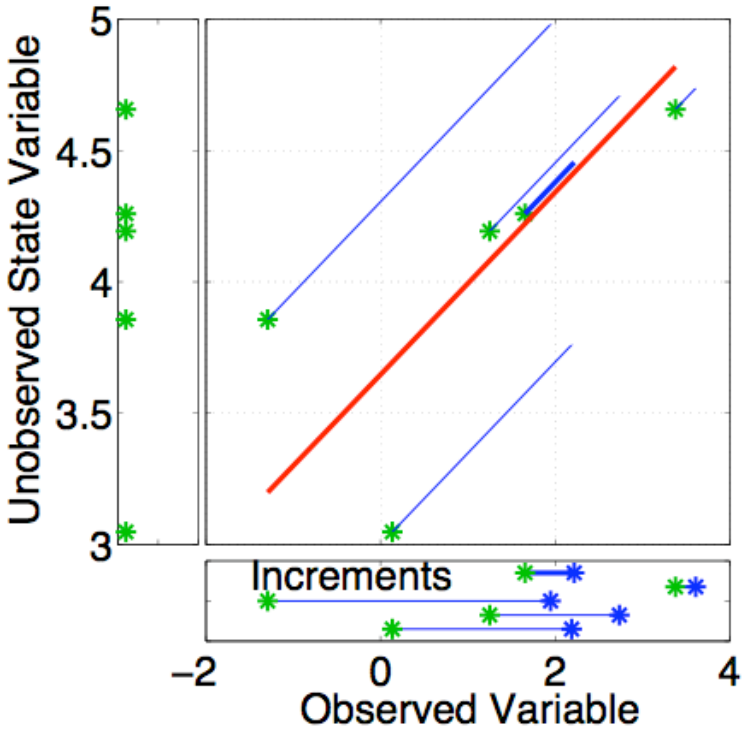


Have joint prior distribution of two variables.

Next, regress the observed variable increments onto increments for the unobserved variable.

Equivalent to first finding image of increment in joint space.

Ensemble filters: Updating additional prior state variables

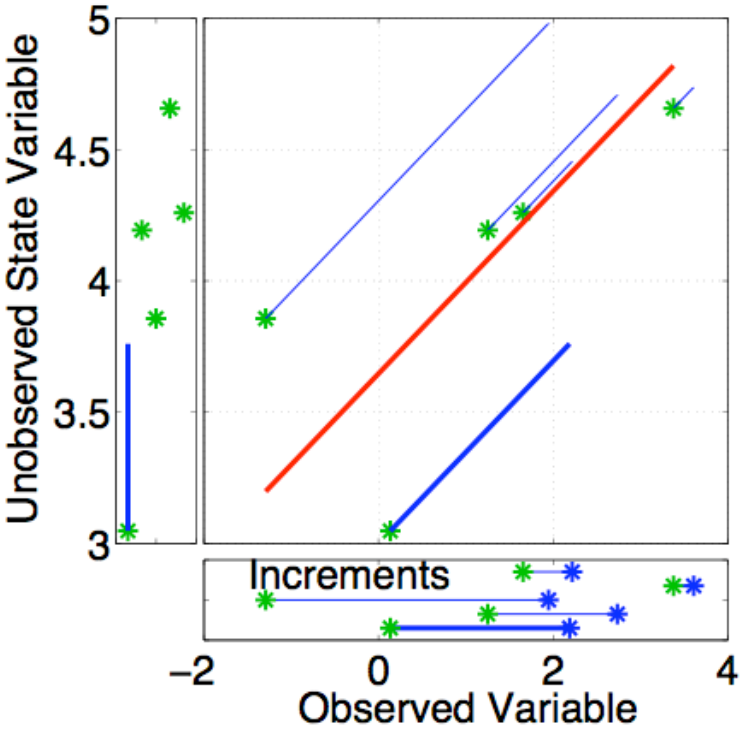


Have joint prior distribution of two variables.

Next, regress the observed variable increments onto increments for the unobserved variable.

Equivalent to first finding image of increment in joint space.

Ensemble filters: Updating additional prior state variables

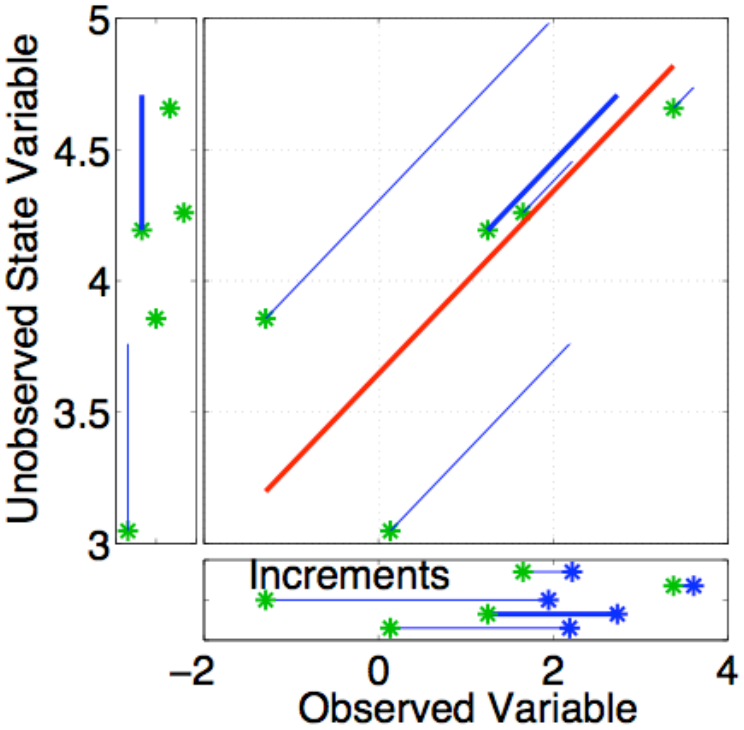


Have joint prior distribution of two variables.

Regression: Equivalent to first finding image of increment in joint space.

Then projecting from joint space onto unobserved priors.

Ensemble filters: Updating additional prior state variables

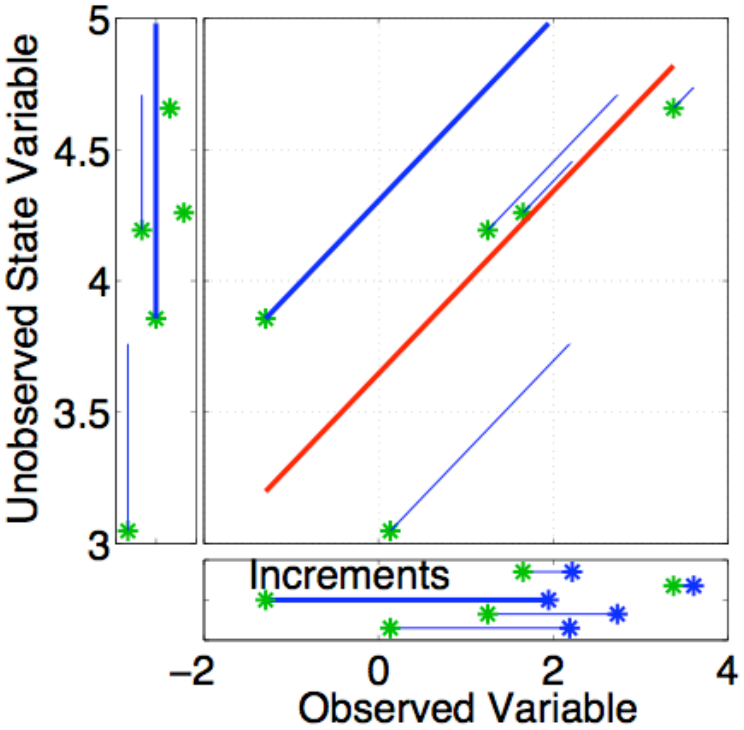


Have joint prior distribution of two variables.

Regression: Equivalent to first finding image of increment in joint space.

Then projecting from joint space onto unobserved priors.

Ensemble filters: Updating additional prior state variables

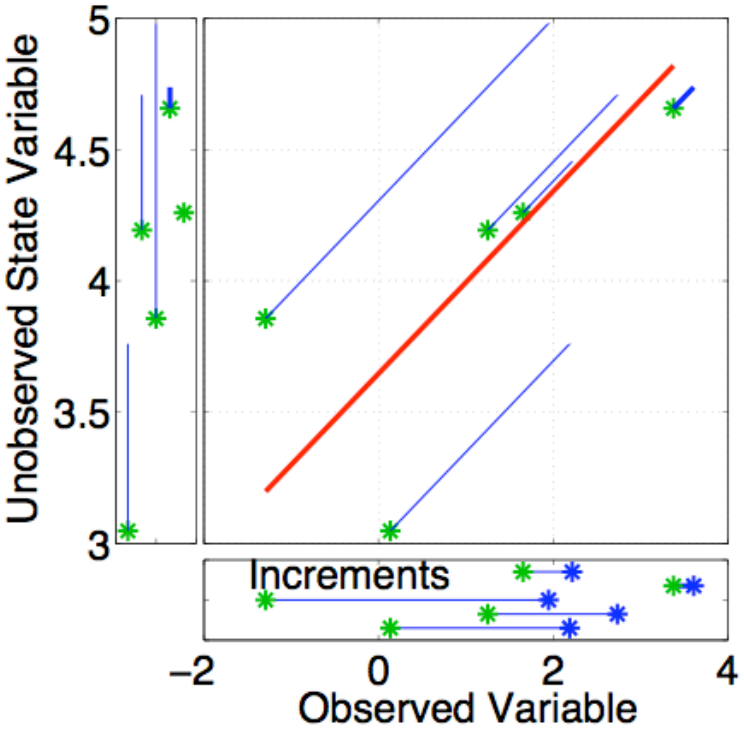


Have joint prior distribution of two variables.

Regression: Equivalent to first finding image of increment in joint space.

Then projecting from joint space onto unobserved priors.

Ensemble filters: Updating additional prior state variables

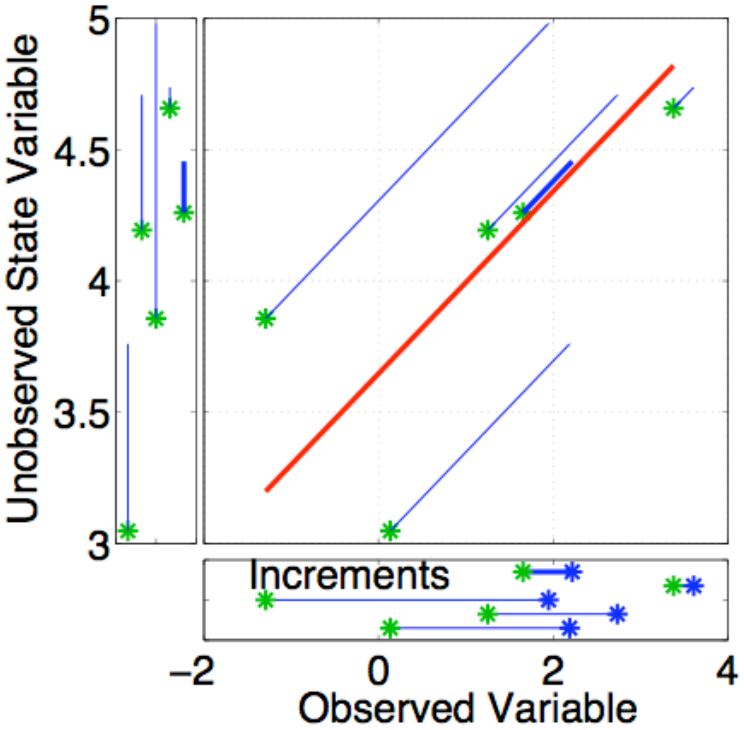


Have joint prior distribution of two variables.

Regression: Equivalent to first finding image of increment in joint space.

Then projecting from joint space onto unobserved priors.

Ensemble filters: Updating additional prior state variables

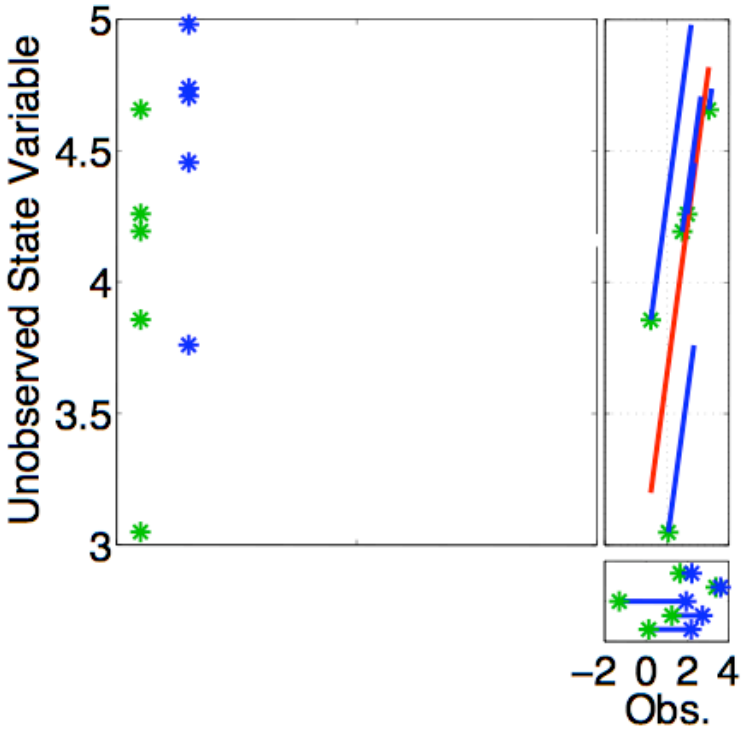


Have joint prior distribution of two variables.

Regression: Equivalent to first finding image of increment in joint space.

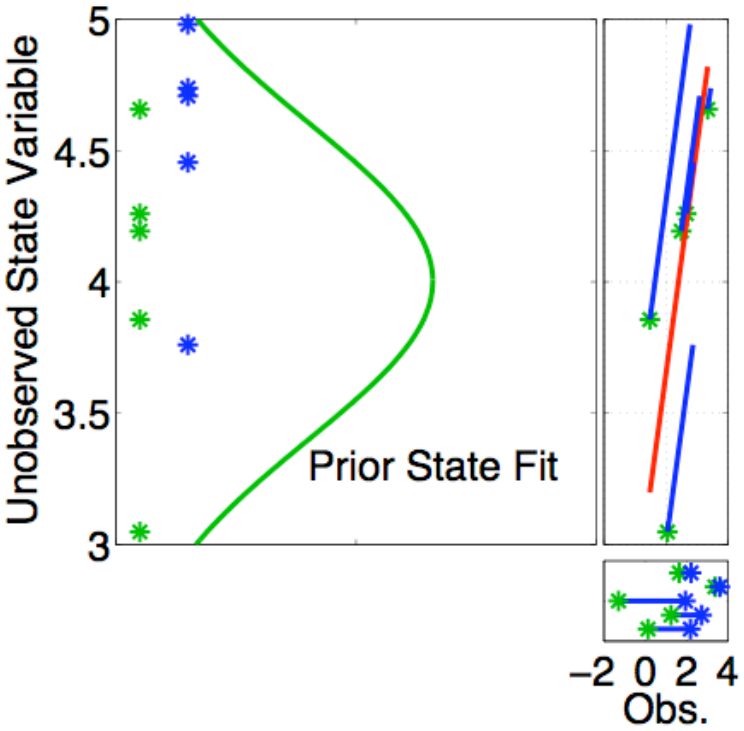
Then projecting from joint space onto unobserved priors.

Ensemble filters: Updating additional prior state variables



Now have an updated (posterior) ensemble for the unobserved variable.

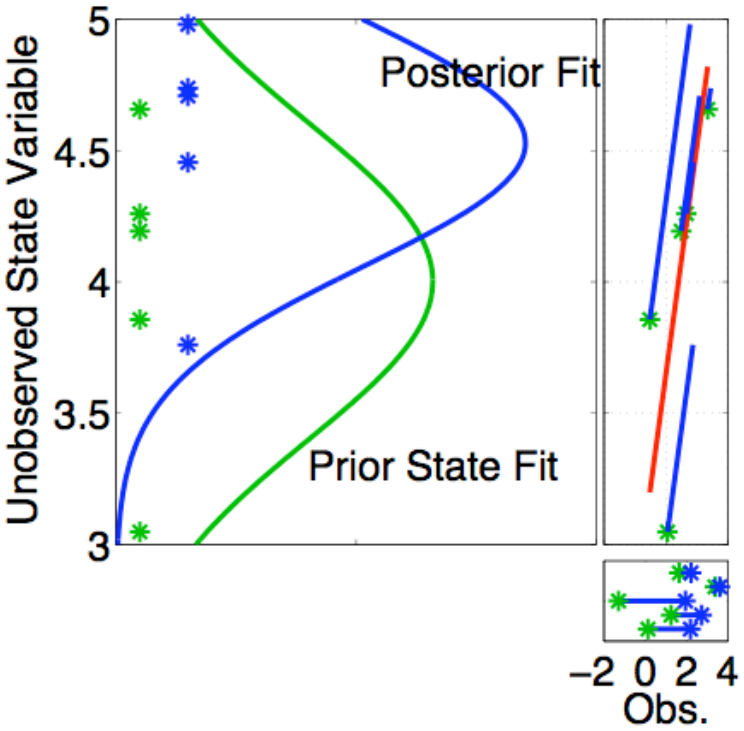
Ensemble filters: Updating additional prior state variables



Now have an updated (posterior) ensemble for the unobserved variable.

Fitting Gaussians shows that mean and variance have changed.

Ensemble filters: Updating additional prior state variables



Now have an updated (posterior) ensemble for the unobserved variable.

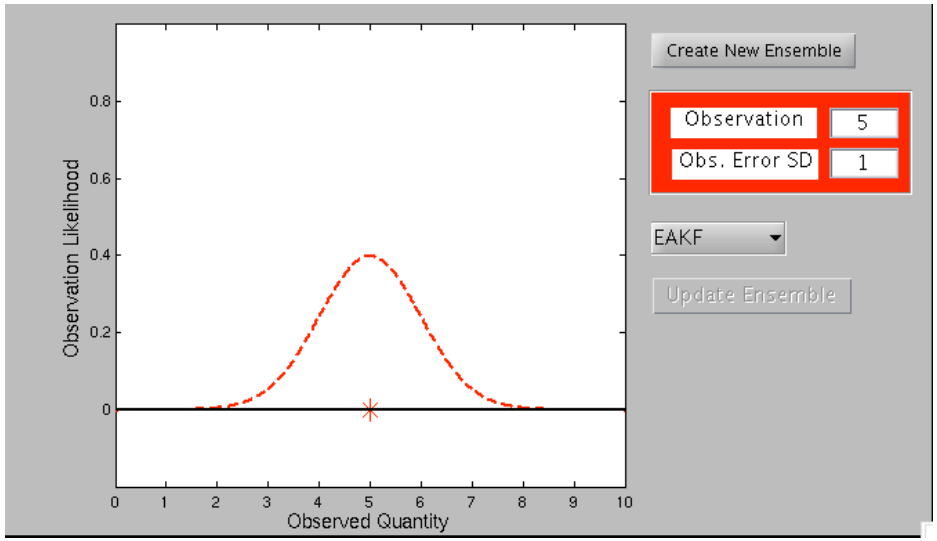
Fitting Gaussians shows that mean and variance have changed.

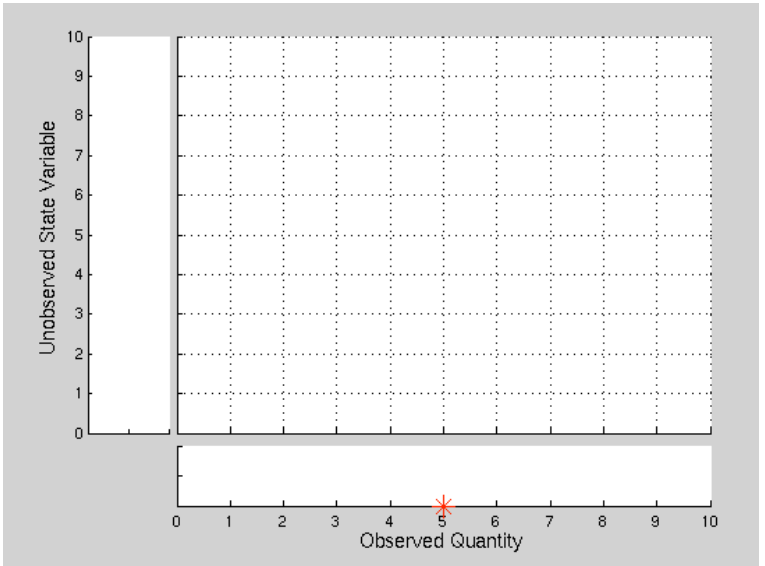
Other features of the prior distribution may also have changed.

Matlab Hands-On: twod_ensemble

Purpose:

Explore how an unobserved state variable is updated by an observation of another state variable.





Matlab Hands-On: twod_ensemble

Procedure:

This script opens two windows: a menu window and a diagnostic window.

1. Use the red dialog boxes to set the observation likelihood mean and standard deviation.
2. To create a prior ensemble:
   a. Select Create New Ensemble .
   b. Click on large white area in the diagnostic window to create an ensemble member. The horizontal coordinate is the value of the observed state variable and the vertical coordinate is the corresponding unobserved variable. Repeat a few times.
   c. Click on a gray area of the diagnostic window to finish ensemble.
3. Select Update Ensemble to see the updated ensemble.
4. The pull-down menu allows you to select different ensemble filter algorithms.

Matlab Hands-On: twod_ensemble

What do I see?

The GUI window displays the prior, observation, and posterior for the observed variable only.

The diagnostic window has three panels. The large central panel is a plot of the joint distribution of the unobserved and observed state variable. The lower panel is the marginal distribution for just the observed variable and corresponds to part of what is shown in the GUI window. The left panel is the marginal for the unobserved variable. It shows how the unobserved variable is updated by the observation.

Matlab Hands-On: twod_ensemble

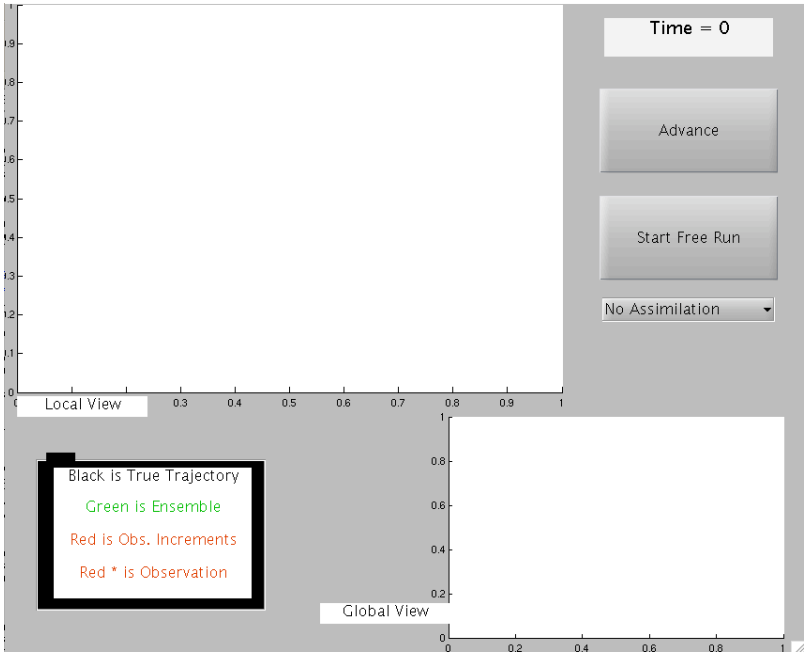
Explorations:

1. Create ensemble members that are nearly on a line. Look at the impact of the different ensemble update algorithms. Remember that the EnKF is stochastic so it gives different answers when tried repeatedly for the same prior.
2. What happens for nearly uncorrelated observed and unobserved variables? Create a roundish cloud of points for the prior.
3. What happens with a bimodal distribution?
4. Try prior ensembles with various types of outliers.
5. Can you see systematic differences in the different ensemble algorithms?

Matlab Hands-On: run_lorenz_63

Purpose:

Explore behavior of ensemble filters in a low-order, chaotic dynamical system, the Lorenz 1963 model.



Matlab Hands-On: run_lorenz_63

Procedure:

1. To see individual model advance and assimilation steps, select the top button on the menu window (it will alternate between Advance Model and Assimilate Obs).
2. Selecting Start Free Run starts a sequence of advance and assimilation steps.
3. Selecting Stop Free Run stops the sequence of steps.
4. The ensemble Kalman filter algorithm can be changed with the pull-down.
   Selecting No Assimilation lets the truth and the model run without assimilation.

Matlab Hands-On: run_lorenz_63

What do I see?

Both panels show the time evolution of the true state (black) and 20 ensemble members (green).

Red asterisks mark the observation and red segments show the increments for the ensemble members.

The smaller panel shows a global view of the true trajectory.

The smaller window shows a magnified view of the recent evolution of the truth and the ensemble.

At each observation time, the three components of the truth are 'observed' by adding a random draw from a standard normal distribution to the true value.

Matlab Hands-On: run_lorenz_63

Explorations:

1. Select Start Free Run and watch the evolution of the ensemble. Try to understand how the ensemble spreads out.

2. Restart the GUI and select an ensemble filter. Do individual advances and assimilations and observe the behavior.

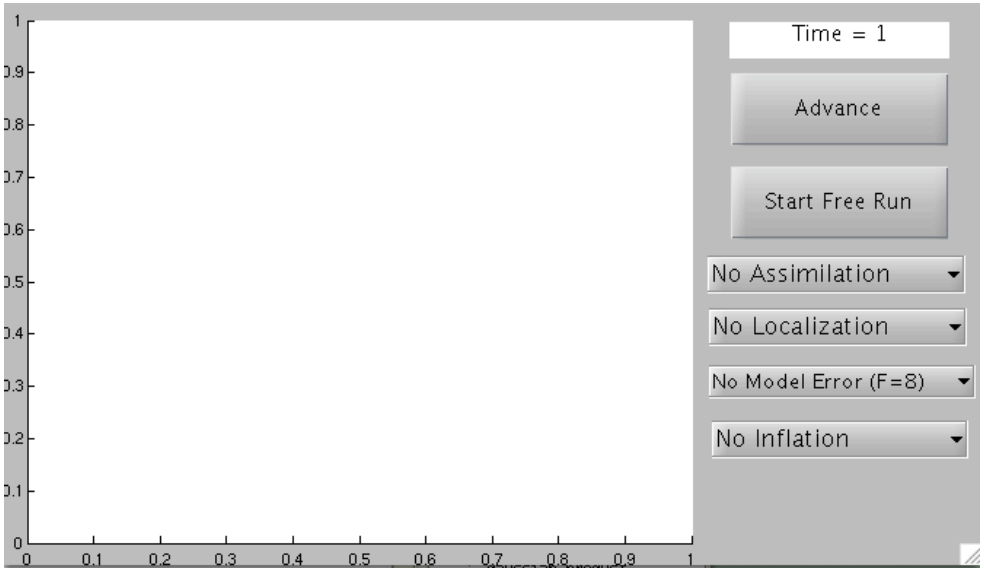
3. Do some free runs with assimilation turned on.

4. Can you identify qualitative differences in the behavior of the different assimilation algorithms?

Matlab Hands-On: run_lorenz_96

Purpose:

Explore the behavior of ensemble filters in a 40-variable chaotic dynamical system; the Lorenz 1996 model. Understand the need for localization and inflation in ensemble assimilation. Explore the impact of model error on assimilations.



Matlab Hands-On: run_lorenz_96

Procedure:

1. To see individual model advance and assimilation steps, select the top button on the menu window (it will alternate between Advance Model and Assimilate Obs).
2. Selecting Start Free Run starts a sequence of advance and assimilation steps.
3. Selecting Stop Free Run stops the sequence of steps.
4. The ensemble Kalman filter algorithm can be changed with the pull-down. Selecting No Assimilation lets the truth and the model run without assimilation.
5. Localization can be set to none or 0.3 with a pull-down.
6. Model error can be set to none (forcing = 8) or substantial (forcing = 6) with a pull-down.
7. No inflation or moderate inflation (1.10) can be selected with a pull-down.

Matlab Hands-On: run_lorenz_96

What do I see?

The panel shows the time evolution of the true state (black) and 20 ensemble members (green).

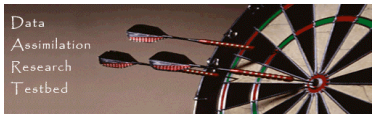
At each observation time, the 40 observations are marked by red asterisks.

Matlab Hands-On: run_lorenz_96

Explorations:

1. Do an extended free run to see error growth in the ensemble. How long does it take to saturate?

2. Turn on assimilation and explore how the different filters impact the ensemble.
3. Turn on localization to see how this can change filter performance.
4. Turn on model error and explore how the ensemble changes.
5. Turn on inflation.
6. Explore other combinations of the filtering options.


Data
Assimilation
Research
Testbed