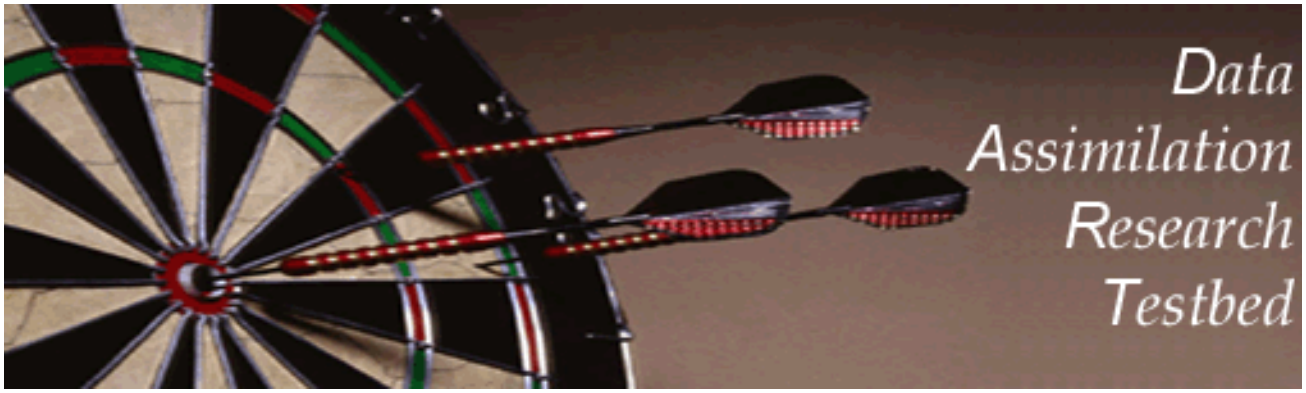
# Data Assimilation Research Testbed Tutorial




## Section 8: Dealing with Sampling Error

Version 1.0: June, 2005

## Ensemble filters: Updating additional prior state variables.

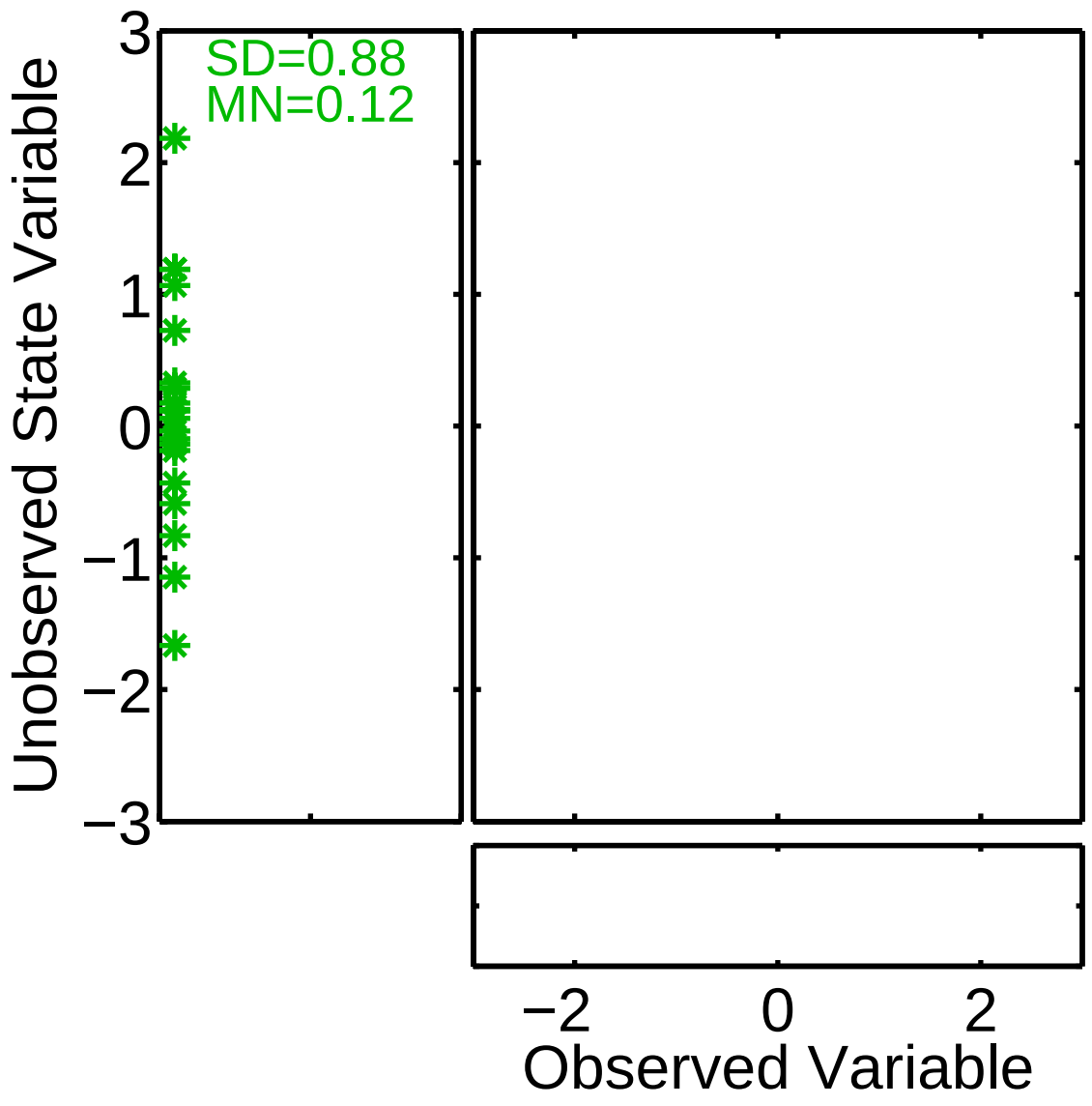
Two primary error sources:

1. Sampling error due to noise.
   Even if linear relation, sample regression coefficient imprecise.

2. Linear approximation is invalid.
   Substantial nonlinearity in 'true' relation over range of prior (see section 10).

May need to address both issues for good performance.

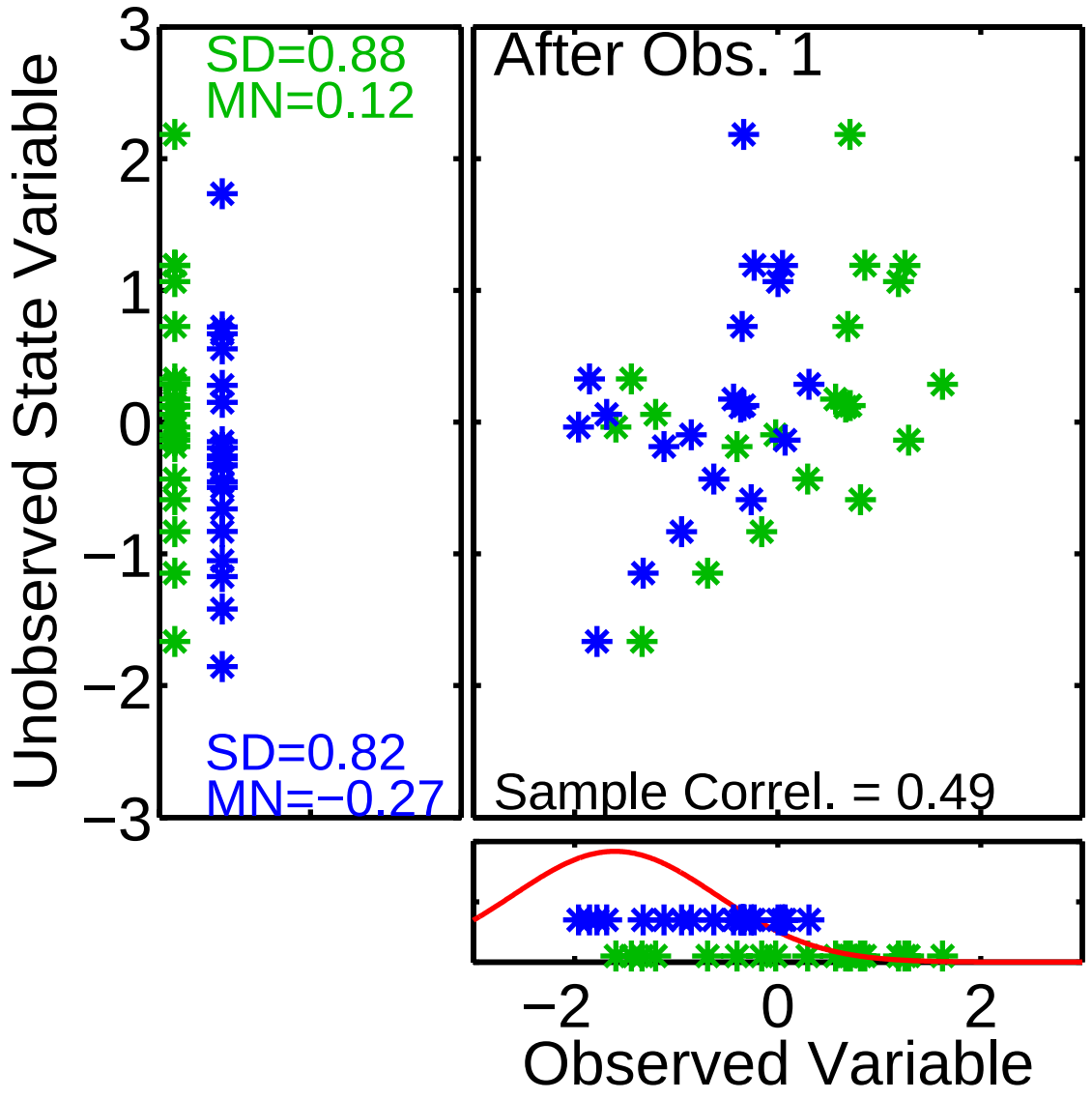
# Regression sampling error and filter divergence




Suppose unobserved state variable is known to be unrelated to set of observed variables.

Unobserved variable should remain unchanged.

# Regression sampling error and filter divergence

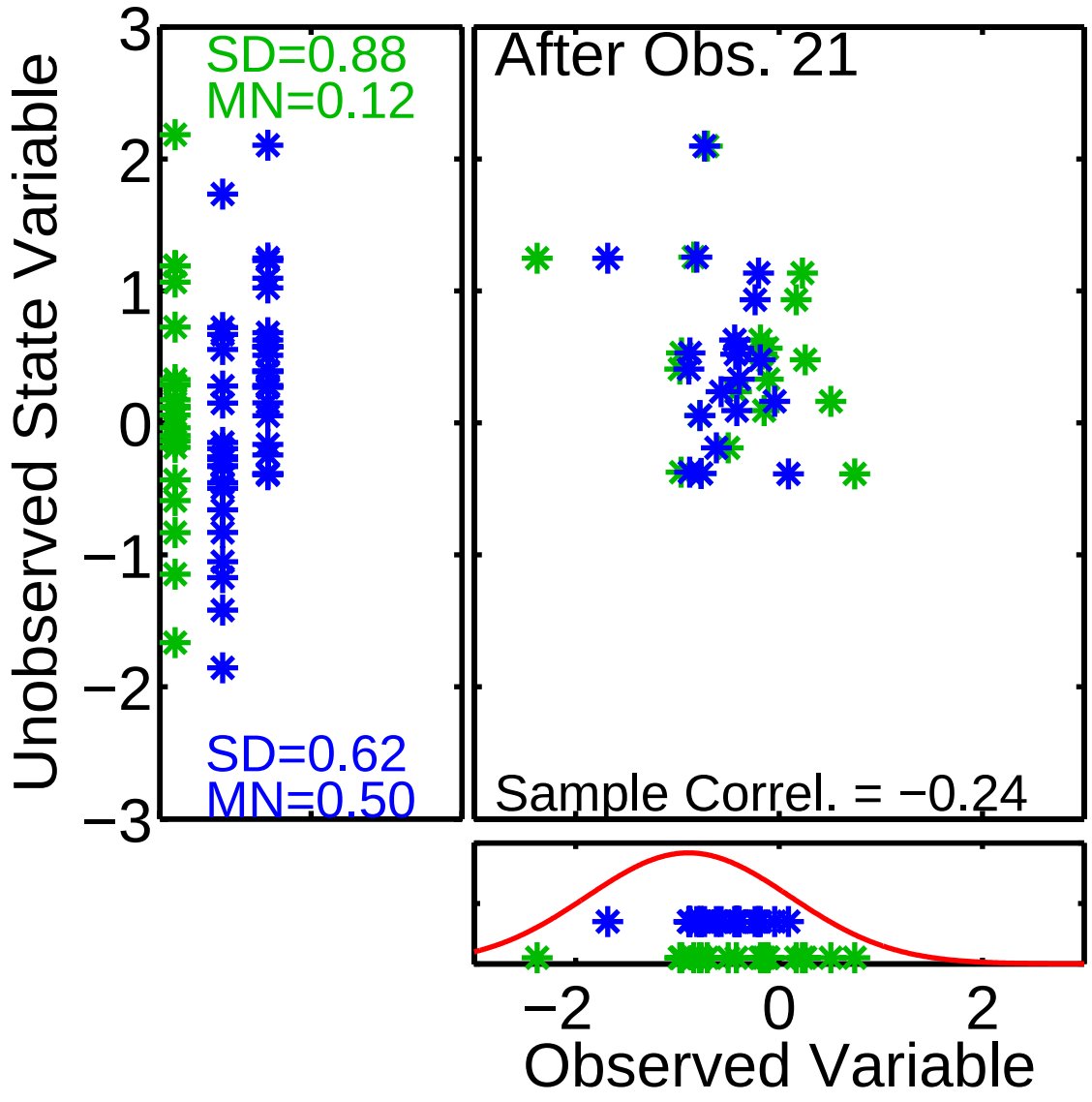



Suppose unobserved state variable is known to be unrelated to set of observed variables.

Finite samples from joint distribution will have non-zero correlation (expected |corr| = 0.19 for 20 samples).

After one observation, unobs. variable mean and S.D. change.

# Regression sampling error and filter divergence

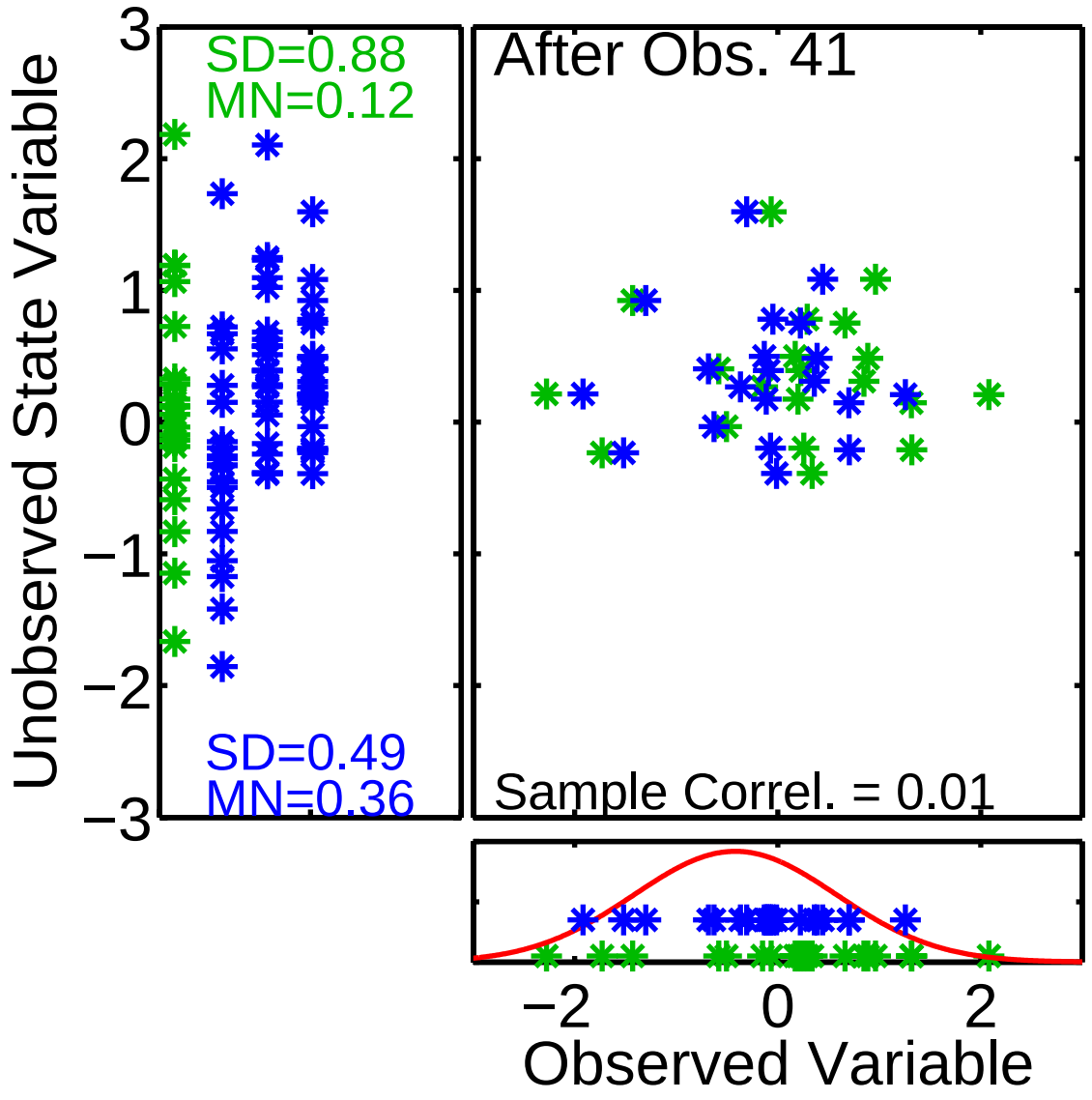



Suppose unobserved state variable is known to be unrelated to set of observed variables.

Unobserved variable should remain unchanged

Unobserved mean follows a random walk as more obs. are used.

# Regression sampling error and filter divergence



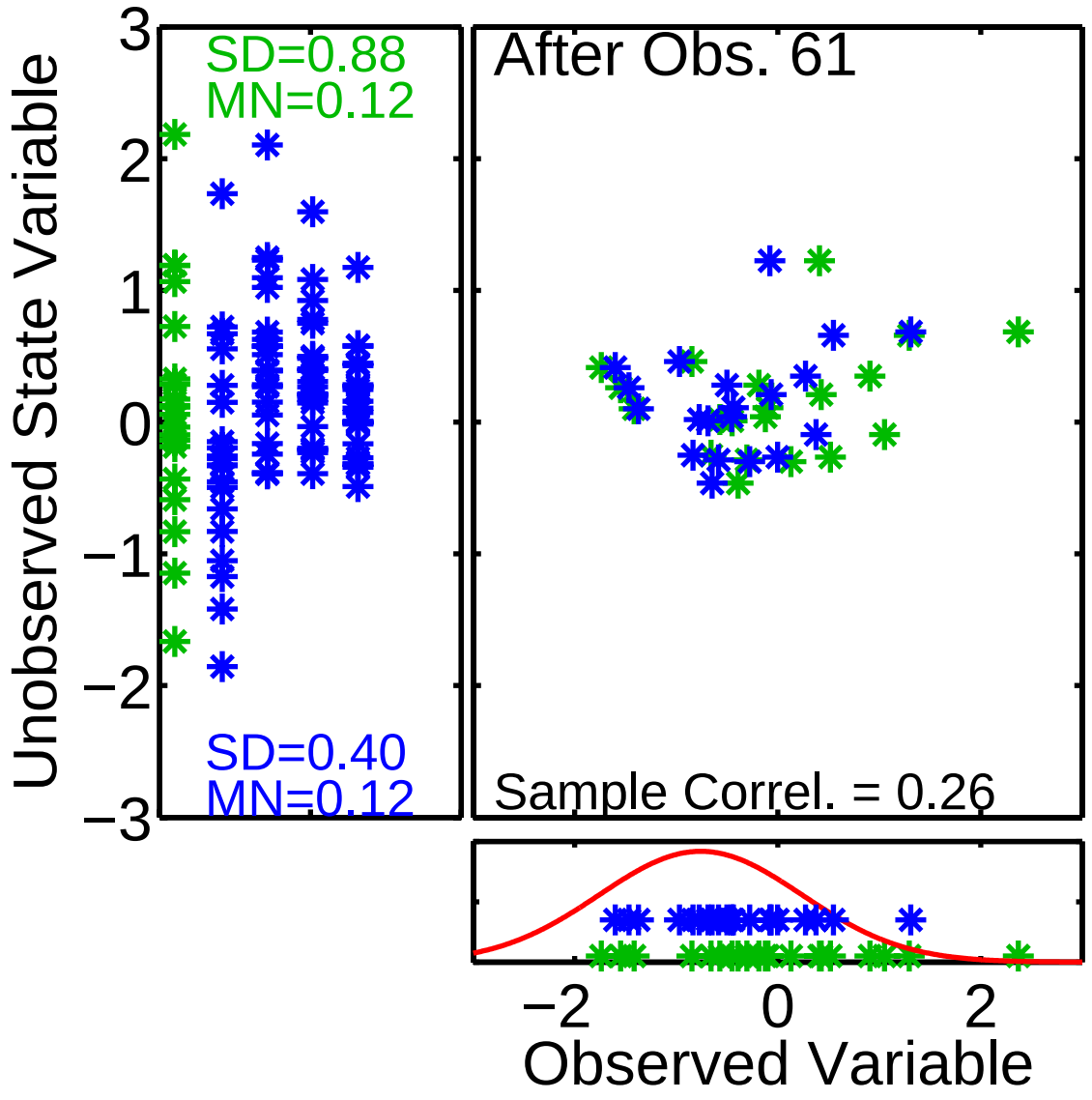

Suppose unobserved state variable is known to be unrelated to set of observed variables.

Unobserved variable should remain unchanged

Unobserved standard deviation is persistently decreased.

Expected change in |SD| is negative for any non-zero sample correlation!

# Regression sampling error and filter divergence



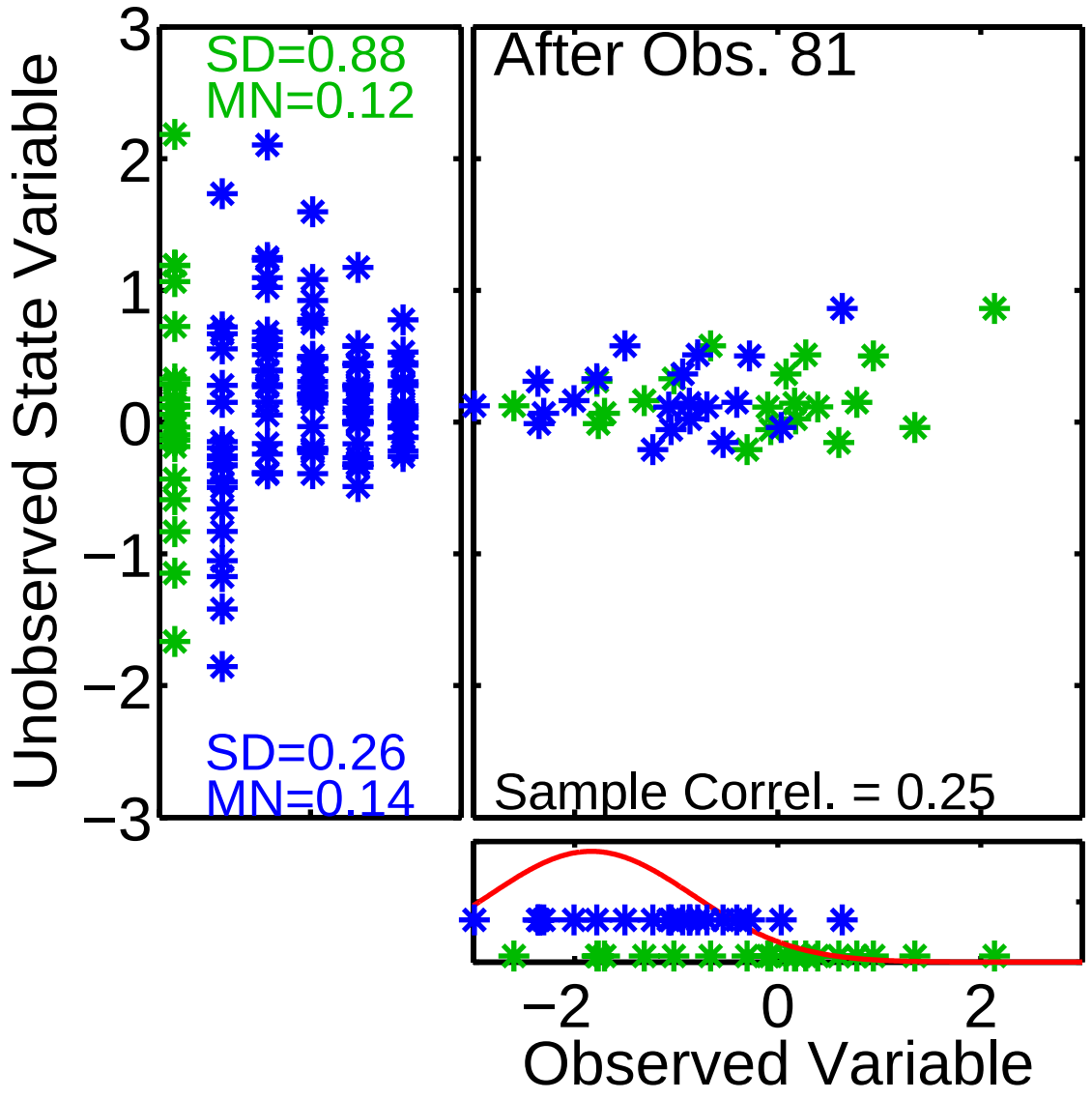

Suppose unobserved state variable is known to be unrelated to set of observed variables.

Unobserved variable should remain unchanged

Unobserved standard deviation is persistently decreased.

Expected change in |SD| is negative for any non-zero sample correlation!

# Regression sampling error and filter divergence



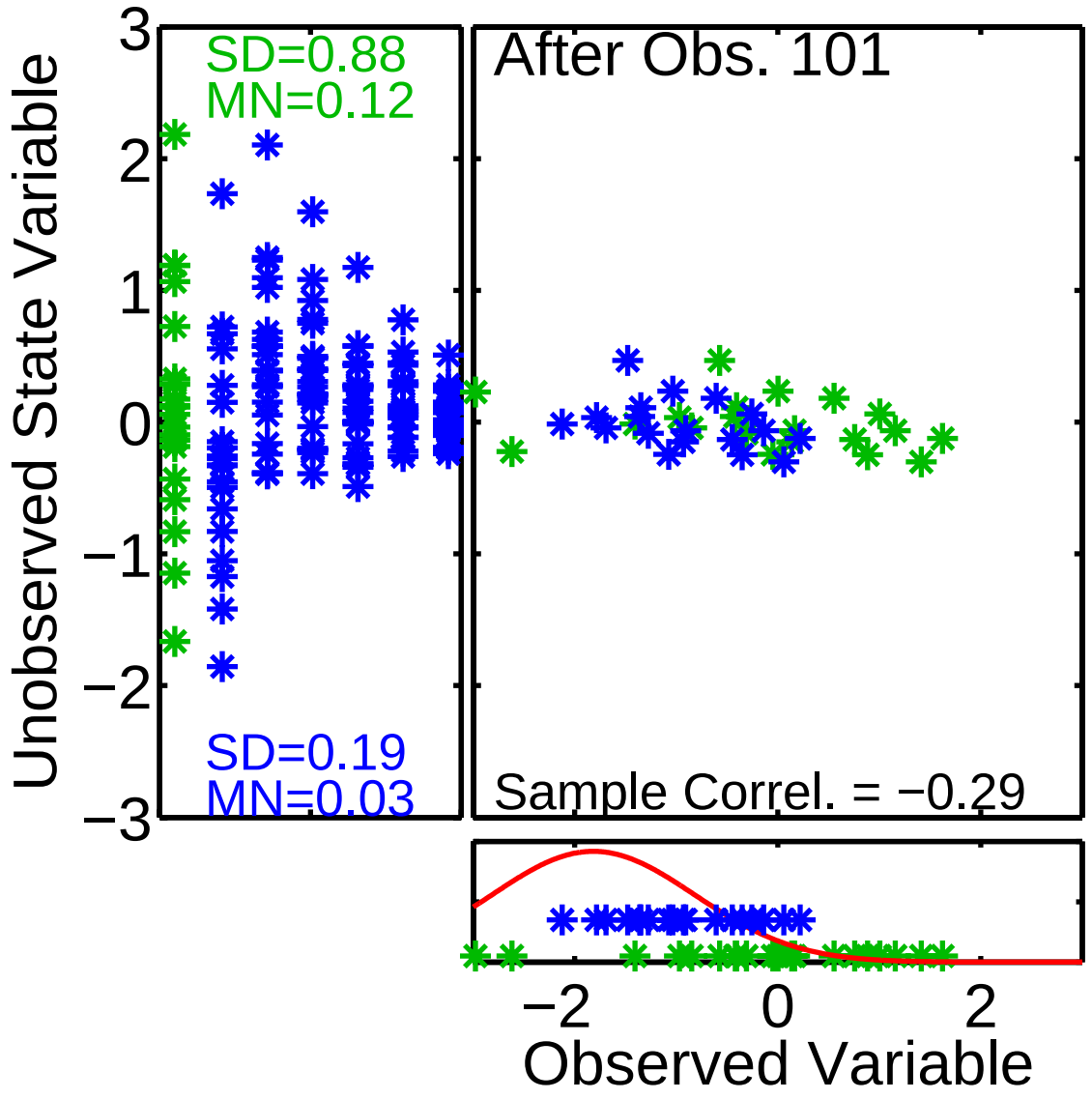

Suppose unobserved state variable is known to be unrelated to set of observed variables.

Unobserved variable should remain unchanged

Unobserved standard deviation is persistently decreased.

Expected change in |SD| is negative for any non-zero sample correlation!

# Regression sampling error and filter divergence




Suppose unobserved state variable is known to be unrelated to set of observed variables.

Estimates of unobs. become too confident

Give progressively less weight to any meaningful observations.

End result can be that meaningful obs. are essentially ignored.

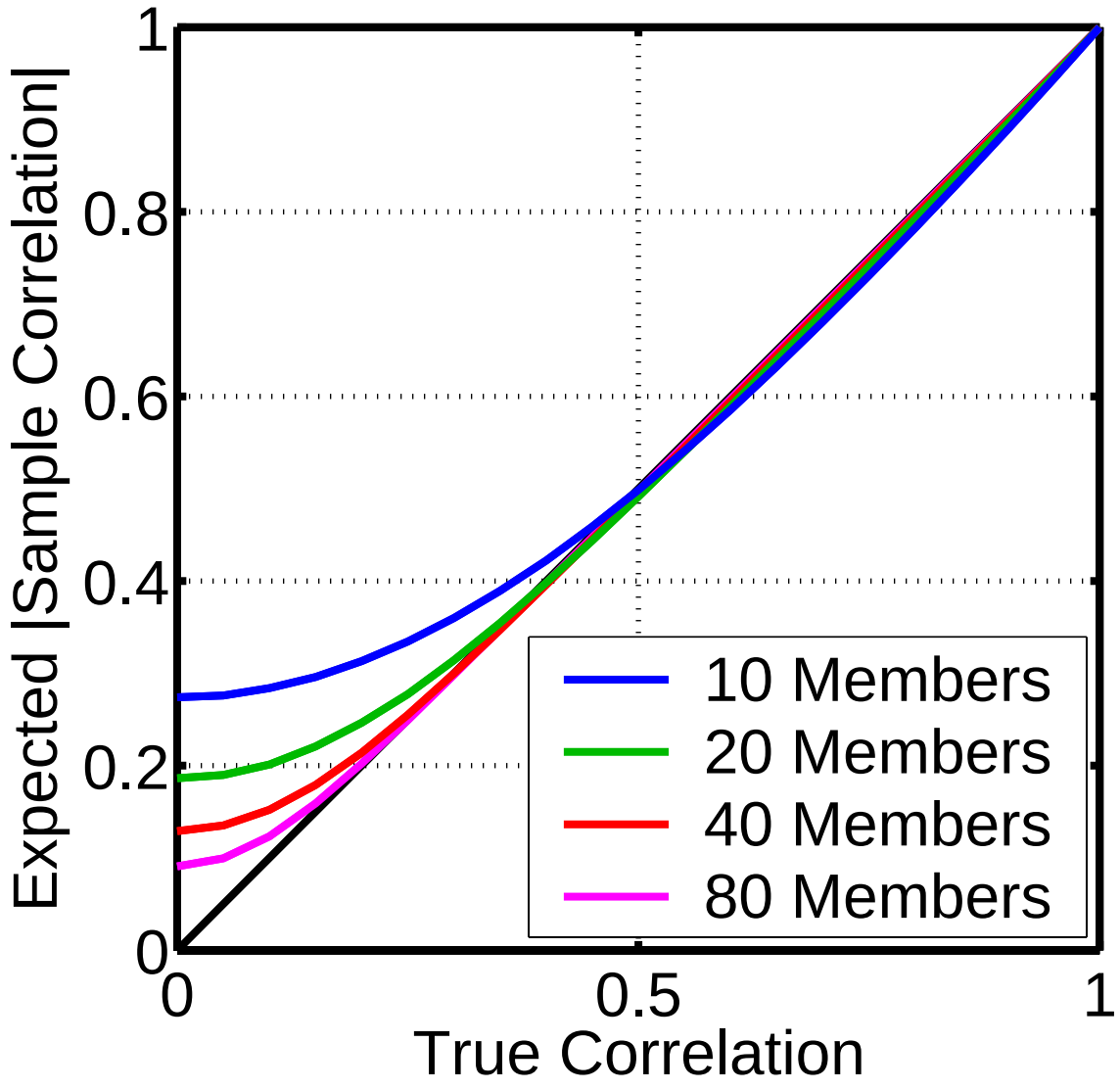
Ignoring meaningful observations due to overconfidence is one type of FILTER DIVERGENCE.

This was seen in the initial Lorenz-96 (40-variable) experiment.

The spread became small => the filter thought it had a good estimate.

The error stayed large because good observations were being ignored.

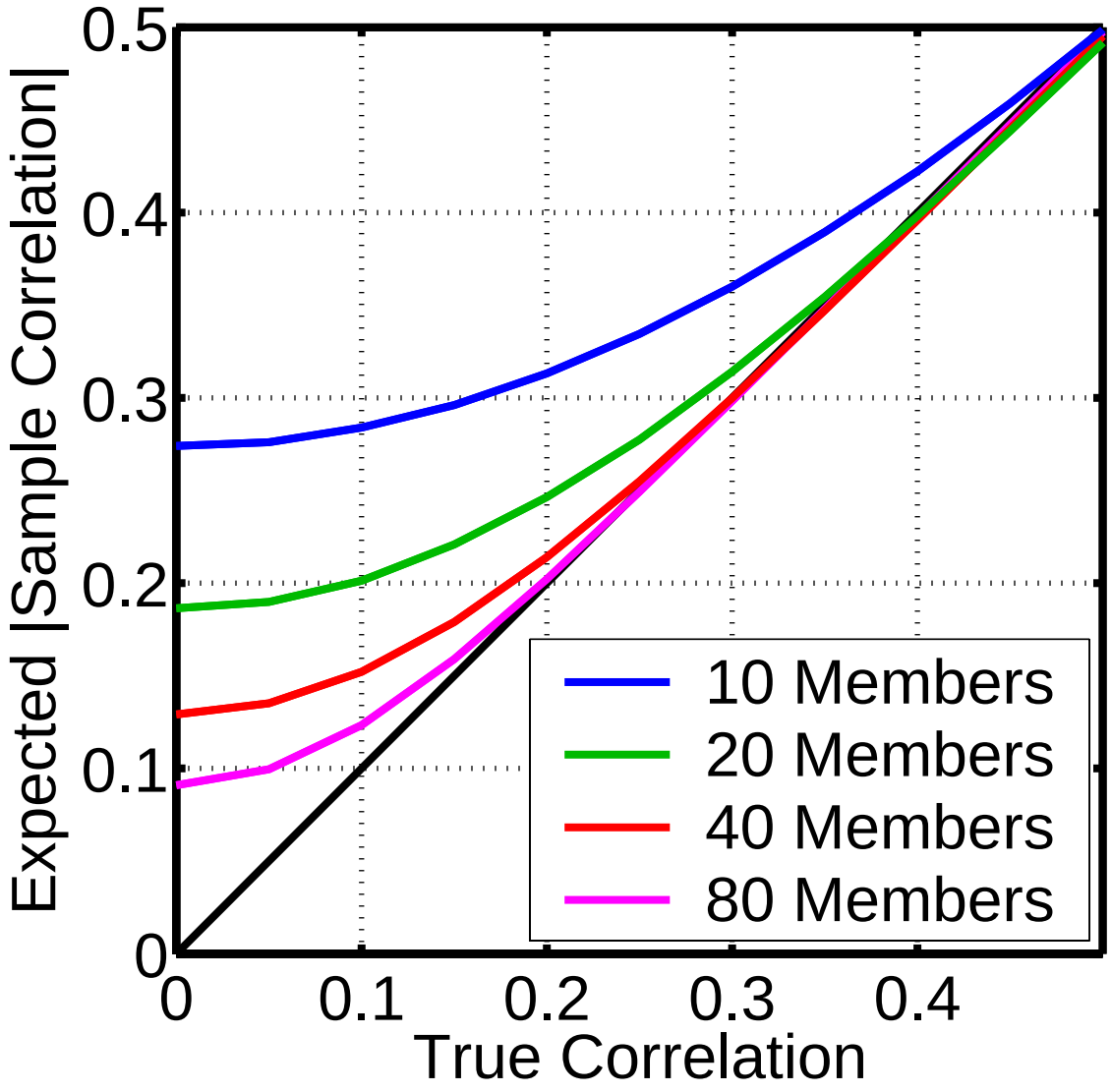
# Regression sampling error and filter divergence




Plot shows expected absolute value of sample correlation vs. true correlation.

Errors decrease with sample size and for large |real correlations|.

# Regression sampling error and filter divergence




Plot shows expected absolute value of sample correlation vs. true correlation.

For negligible true correlations, errors are still significant even for 80 member ensembles.

# Ways to deal with regression sampling error:

1. Ignore it: if number of unrelated observations is small
   and there is some way of maintaining variance in priors.
   (We did this in the 3 and 9 variable models).

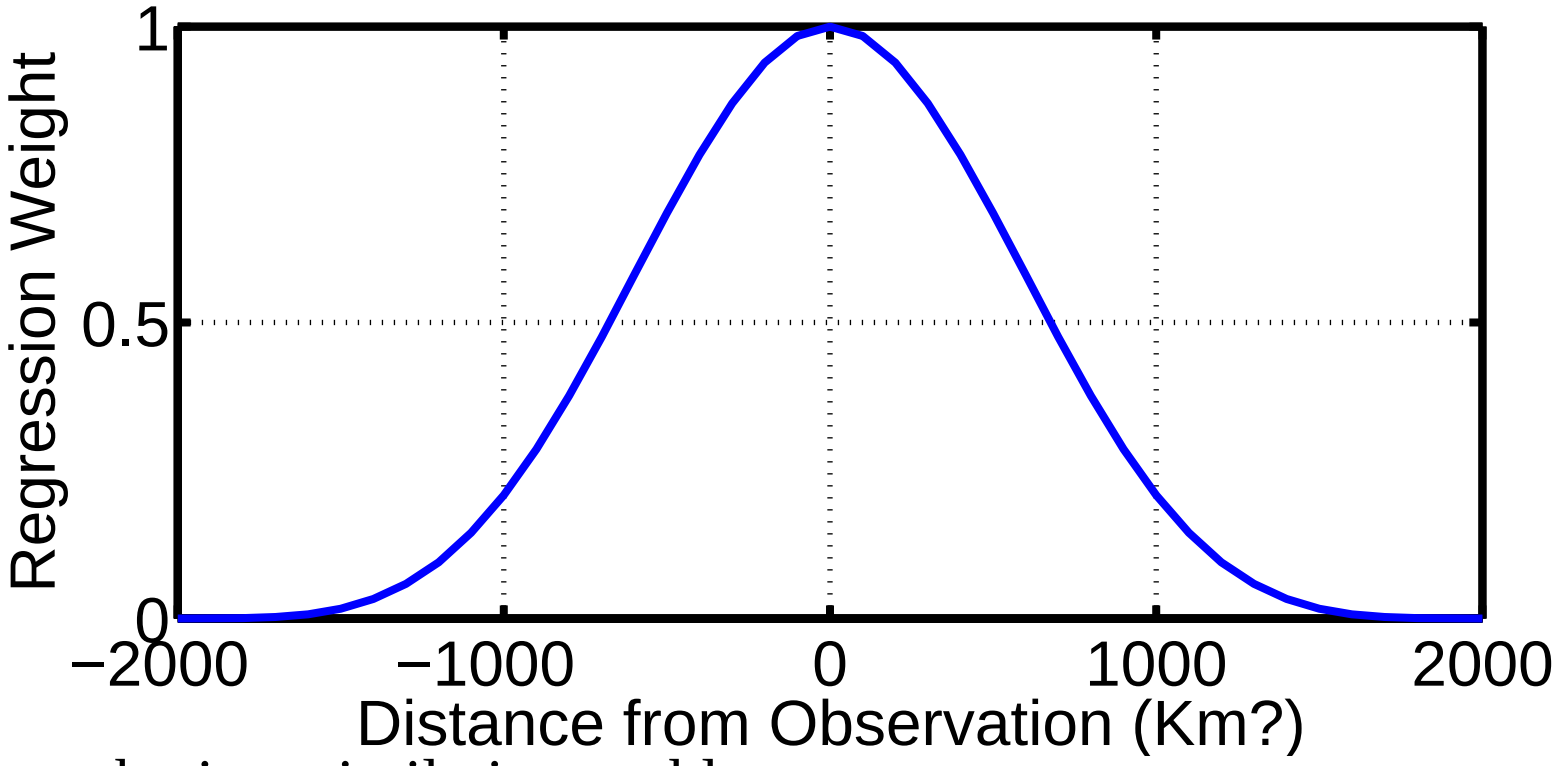
2. Use larger ensembles to limit sampling error.
   (This can get expensive for big problems).
   To try this, modify *ens_size* in *filter_nml* (80 is largest available)

3. Use additional a priori information about relation between
   observations and state variables.
   (Don't let an obs. impact state if they are know to be unrelated)

4. Try to determine the amount of sampling error and correct for it.
   (There are many ways to do this; some are simple, some complex).

# Ways to deal with regression sampling error:

3. Use additional a priori information about relation between observations and state variables.



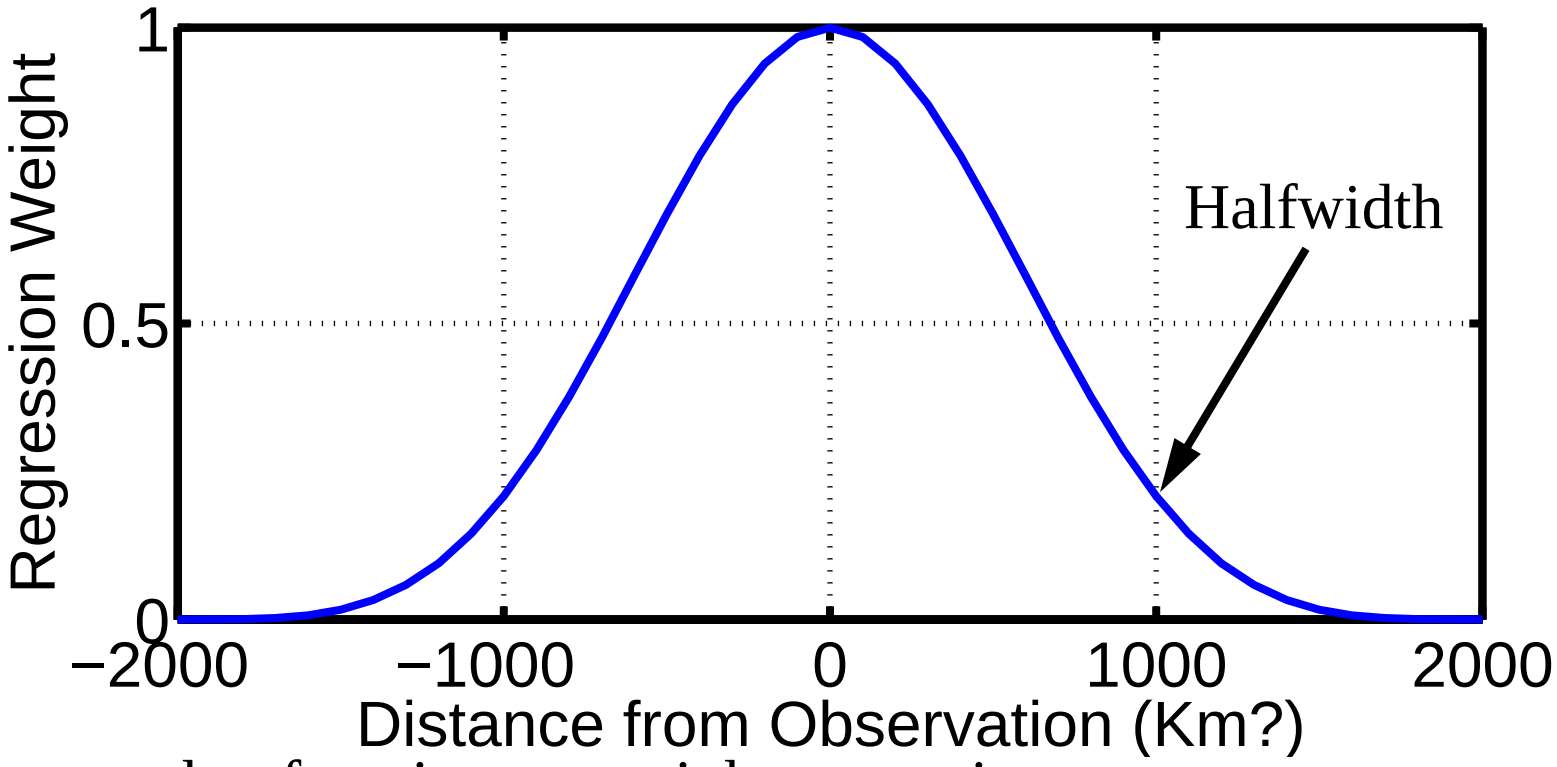

Atmospheric assimilation problems.

Weight regression as function of horizontal *distance* from observation.

Gaspari-Cohn: 5th order compactly supported polynomial.

# Ways to deal with regression sampling error:

3. Use additional a priori information about relation between observations and state variables.




Can use other functions to weight regression.

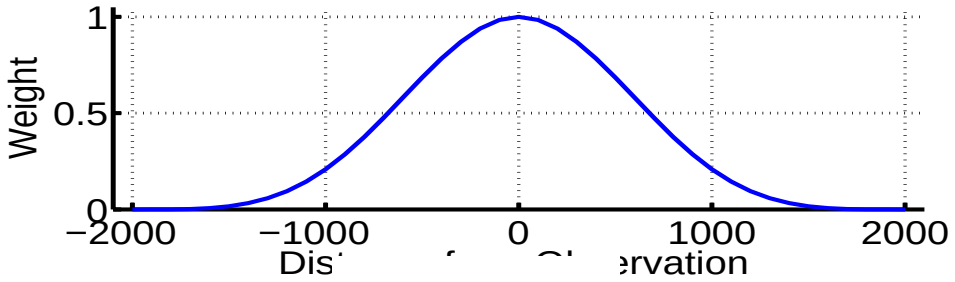
Unclear what *distance* means for some obs./state variable pairs.

Referred to as LOCALIZATION.

DART provides several localization options:

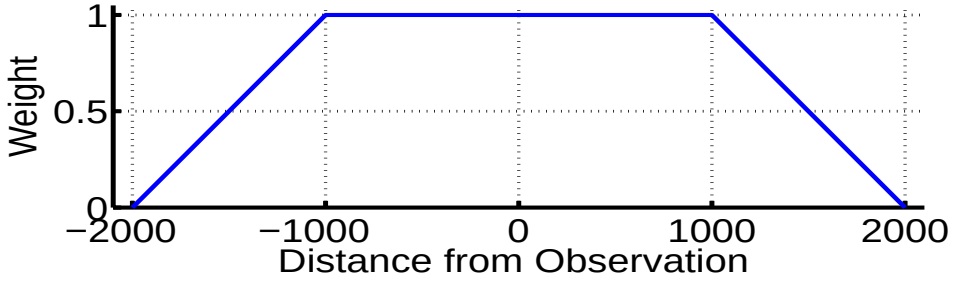
1. Different shapes for the localization function are available.
   Controlled by *select_localization* in *cov_cutoff_nml*.

1=> Gaspari-Cohn




2=>Boxcar


3=>Ramped Boxcar




2. Halfwidth of localization function set by *cutoff* in *assim_tools_nml*

Experimenting with Lorenz-96:

The Lorenz-96 domain is mapped to a [0, 1] periodic range.

Try a variety of half widths for a Gaspari Cohn localization.
(Change *cutoff* in *assim_tools_nml*)

We already know that a very large localization half-width diverges.

What happens for a very small value?

What happens with intermediate values?

Can also try changing the shape:
Try option 2 or 3 for *select_localization* in *cov_cutoff_nml*.

## Ways to deal with regression sampling error:

4. Try to determine the amount of sampling error and correct for it.

Many ways to do this. DART implements one naive way:

1. Take set of increments from a given observation,
2. Suppose this observation and a state variable are not correlated,
3. Compute the expected decrease in spread given not correlated,
4. Add this amount of spread back into the state variable.

The expected decrease in spread is computed by off-line Monte Carlo. Results of off-line simulation are tabulated and applied.

(This can be a very useful technique when you're analytically clueless).

Try this algorithm: set *spread_restoration* in *assim_tools_nml* to true.