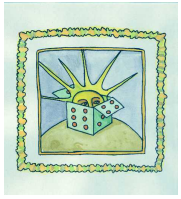
The Matrix Reloaded: Computations for large spatial data sets

Doug Nychka
National Center for Atmospheric Research

- The spatial model
- Solving linear systems
- Matrix multiplication
- Creating sparsity

Sparsity, fast matrix multiplications, iterative solutions.



Supported by the National Science Foundation DMS

The Cast

Reinhard Furrer and Marc Genton

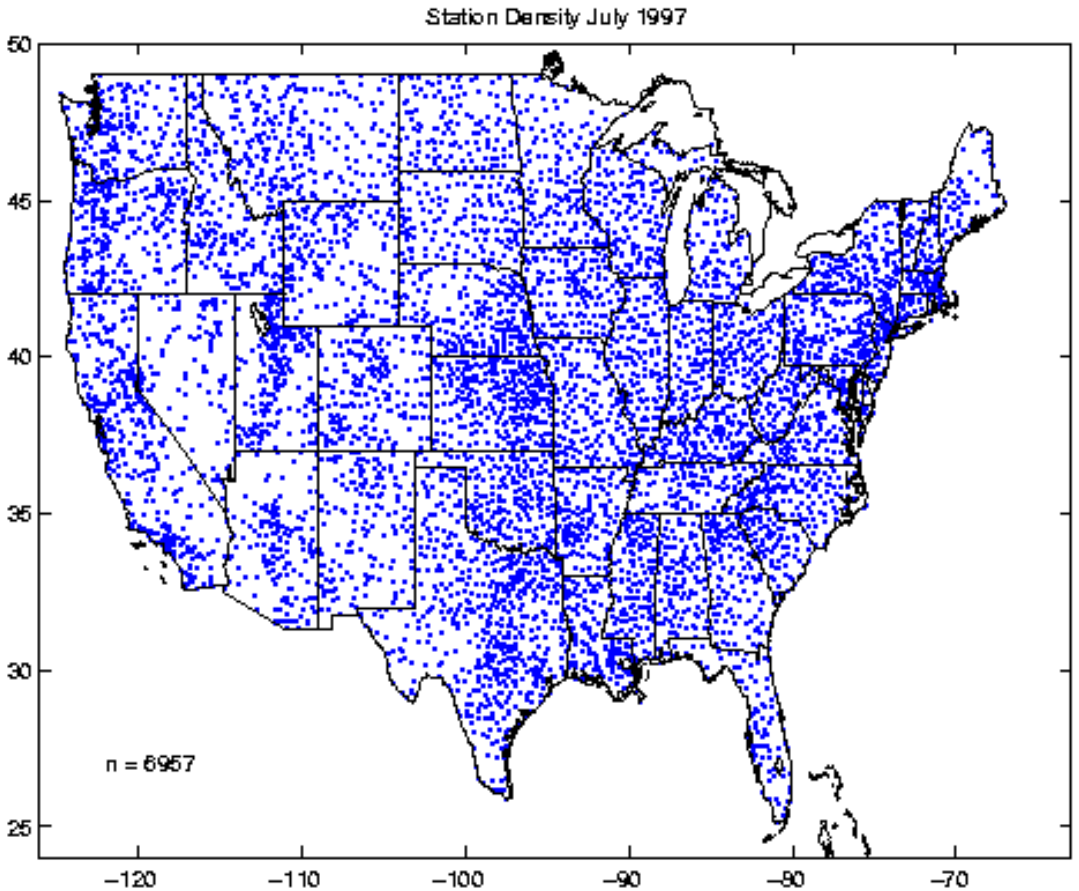
Chris Wikle and J Andrew Royle

Tim Hoar Ralph Milliff and Mark Berliner

Craig Johns

A large spatial dataset

Reporting precipitation stations for 1997.



Spatial Models

We observe a random field, $u(\boldsymbol{x})$, e.g. ozone concentration at location $\boldsymbol{x}$, with covariance function

$$k(\boldsymbol{x}, \boldsymbol{x}') = COV(u(\boldsymbol{x}), u(\boldsymbol{x}')$$

There are other parts of u that are important:

- $E(u(\boldsymbol{x}))$, fixed effects and covariates
- $u(\boldsymbol{x})$ is not Gaussian
- Copies of $u(\boldsymbol{x})$ observed at different times are correlated, e.g ozone fields for each day.

I really don't want to talk about these today!

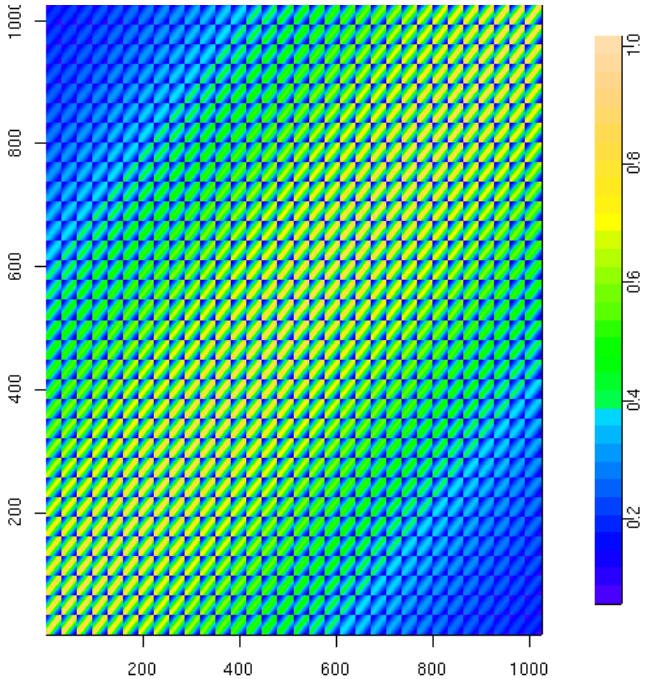
Spatial Models (continued)

Let $\boldsymbol{u}$ be the field values on a large, regular 2-d grid (and stacked as a vector). This is our universe.

$$\Sigma = COV(\boldsymbol{u})$$

The monster matrix Σ

An exponential covariance with range 340 miles for the ozone example.


1000
800
600
400
200
200
400
600
800
1000
1.0
0.8
0.6
0.4
0.2

Observational model

We observe part of $\boldsymbol{u}$, possibly with error.

$$\boldsymbol{Y} = K\boldsymbol{u} + \boldsymbol{e}$$

K is a known "observational functional"'
such as an incidence matrix of ones and zeroes for irregularly spaced data or weighted averages ... or both.

K is usually sparse ...

$COV(\boldsymbol{e}) = R$
(where part of the variability may be due to discretization error.)

Kriging

Assuming $\boldsymbol{Y}$ has zero mean.

$$\hat{\boldsymbol{u}} = COV(\boldsymbol{u}, \boldsymbol{Y})COV(\boldsymbol{Y})^{-1}\boldsymbol{Y}$$

or with $Q = COV(\boldsymbol{Y}) = K\Sigma K^T + R$

$$\hat{\boldsymbol{u}} = \Sigma K Q^{-1}\boldsymbol{Y}$$

and the covariance of the estimate is

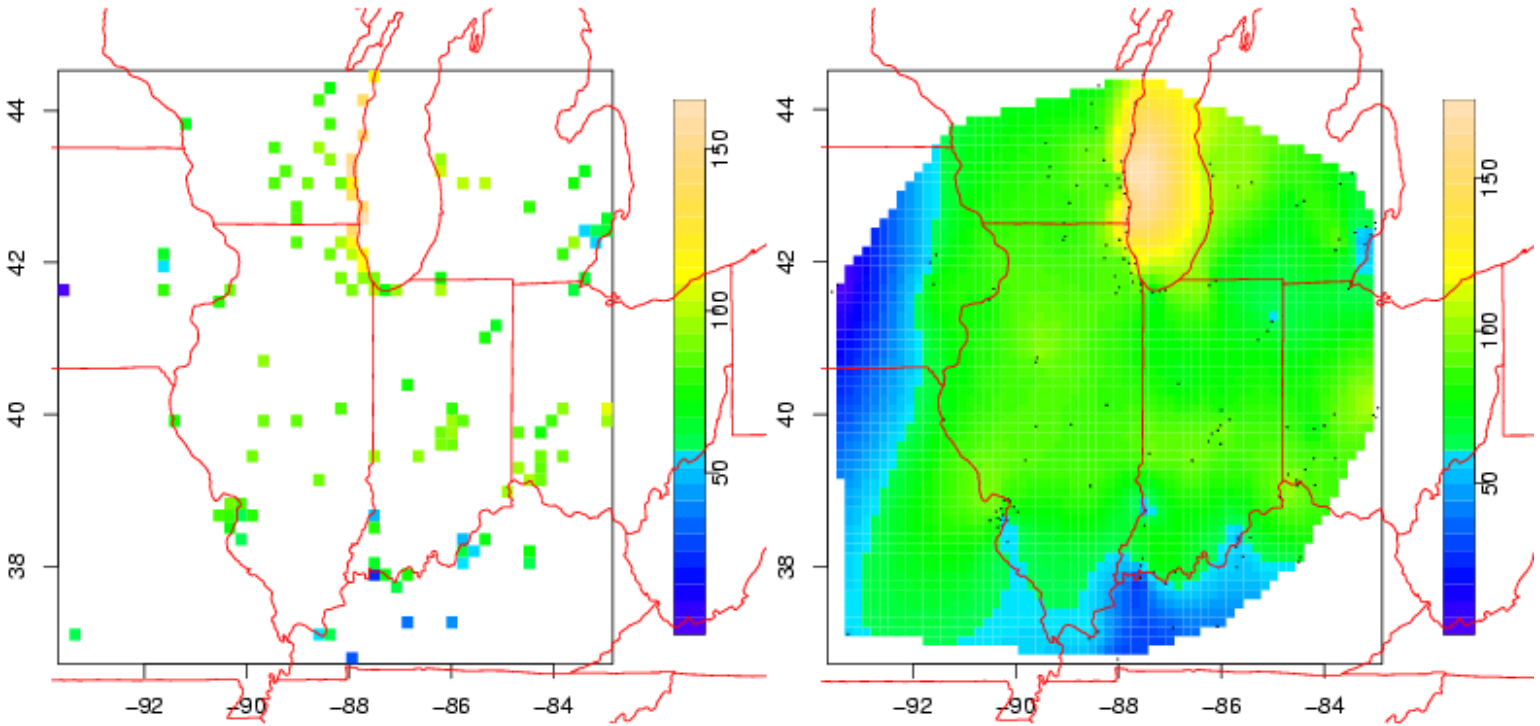
$$\Sigma - \Sigma K Q^{-1} K^T \Sigma$$

I like to think of the estimate as based on the conditional multivariate normal distribution of the grid points given the data: $[\boldsymbol{u}|\boldsymbol{Y}]$

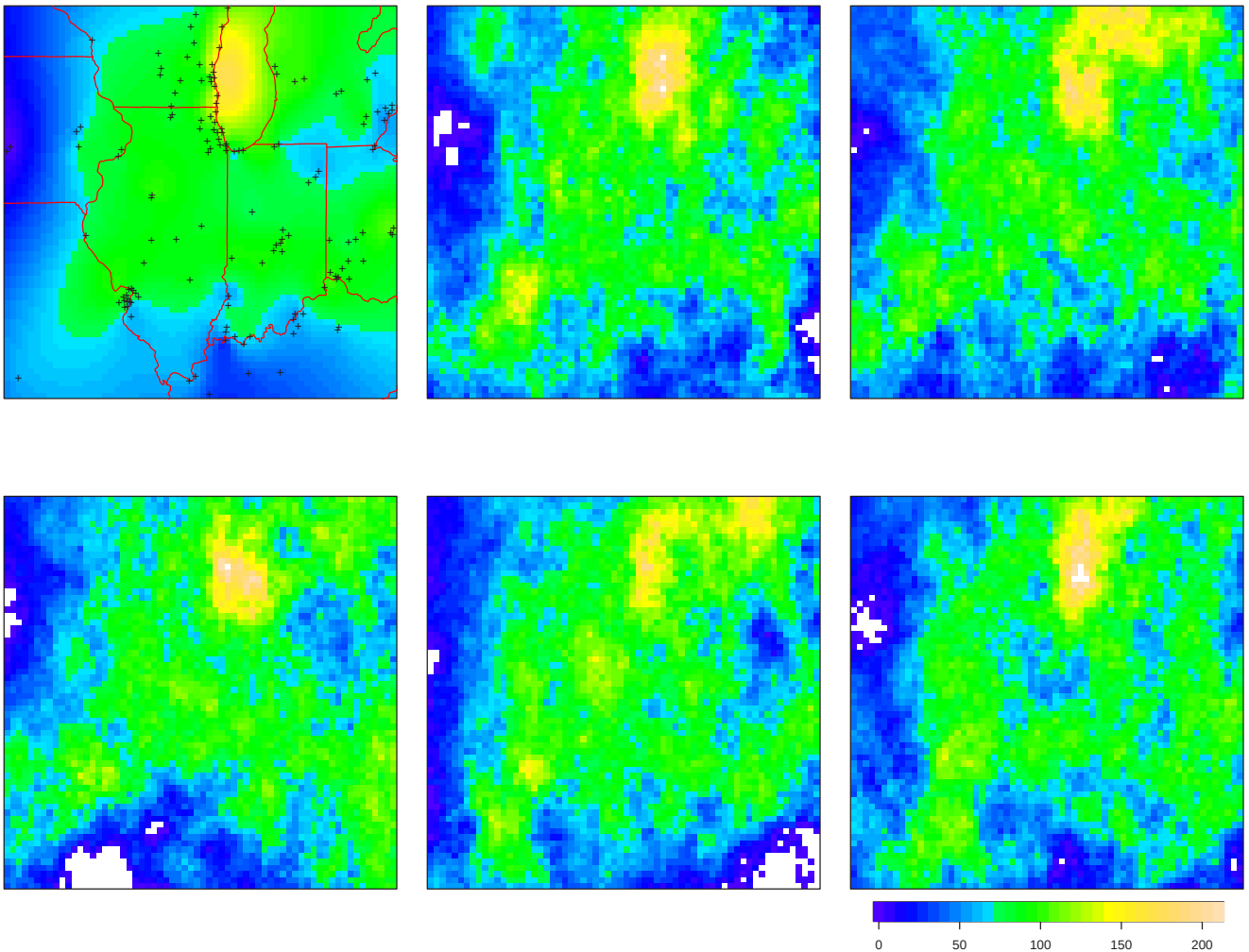
An approximate posterior.

Surface Ozone Pollution (YAOZE)

8-hour average ozone measurements (in PPB) for June 19,1987, Midwest US and the posterior mean surface.



Five samples from the posterior



The problems

Σ is big and so are Q and $\boldsymbol{Y}$!

Simplistic implementations will take too long or involve matrices that are too big.

Some ideas for computation

Don't invert the matrix $Q = (K\Sigma K^T + R)$!

Instead solve the linear system

$$Q\boldsymbol{\omega} = \boldsymbol{Y}$$

for $\boldsymbol{\omega}$ and then

$$\hat{\boldsymbol{u}} = \Sigma K \boldsymbol{\omega}$$

Estimate variability by generating samples from the conditional distribution.

- $\boldsymbol{u}^* \sim N(0, \Sigma)$.
- $\boldsymbol{Y}^* = K\boldsymbol{u}^* + \boldsymbol{e}^*$
- perturbation= $\boldsymbol{u}^* - \Sigma K Q^{-1} \boldsymbol{Y}^*$
- $\hat{\boldsymbol{u}}$ + perturbation

The key third step is just the Kriging estimate!

Solving linear systems

Our job is to find $\boldsymbol{\omega}$ for
$Q\boldsymbol{\omega} = \boldsymbol{Y}$

Conjugate gradient algorithm (CGA) is an iterative method for finding a solution.

- If Q is $n \times n$ it will find the exact solution in n steps but one can stop in much fewer steps to obtain an approximate solution.
- Each iteration only requires two multiplications of Q by a vector.

So CGA never needs to create Q, one only needs to only multiply Q by vectors a limited number of times.

But how do we multiply matrices fast?

Fast multiplication

convolution: If Σ is formed from a stationary covariance matrix then $\Sigma \boldsymbol{v}$ has components

$$\sum_j k(x_i, x_j) v(x_j) = \sum_j \psi(x_i - x_j) v(x_j)$$

so the multiplication is just a convolution to the covariance with the vector (actually an image).

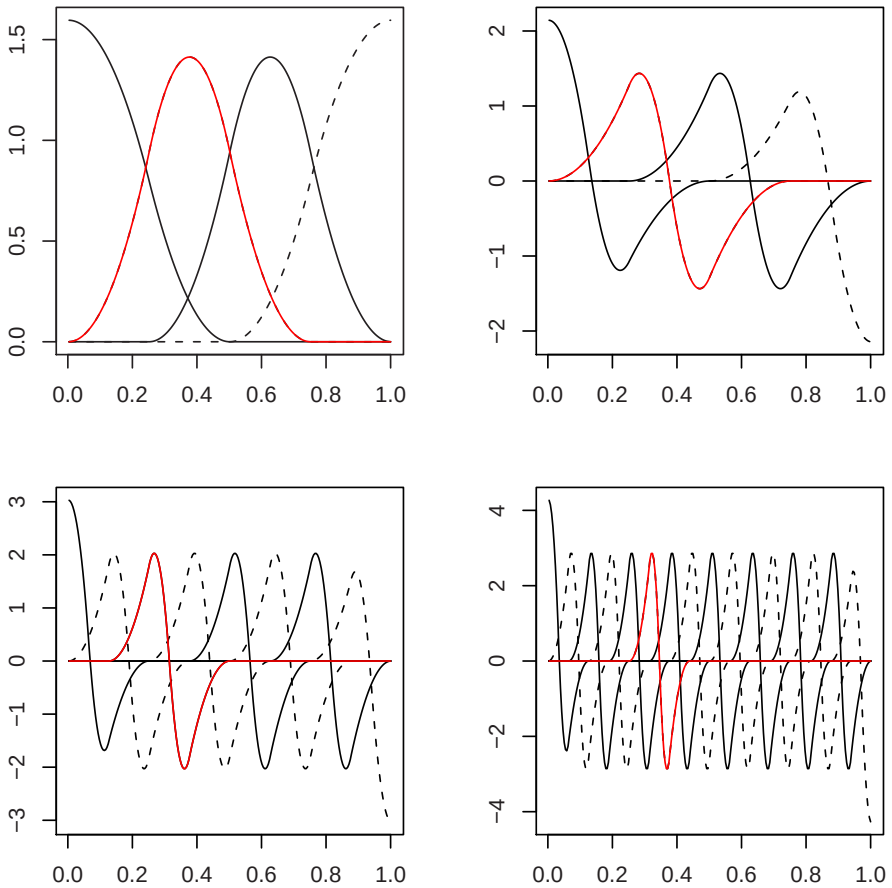
Convolution of an image can be done very quickly using a 2-d fast Fourier transform.

sparsity: If K or K^T are sparse (mostly zeroes) then multiplications can be done quickly by skipping zero elements.

multi-resolution: If $\Sigma = WDW^T$ where $W\boldsymbol{v}$ is the (inverse) discrete wavelet transform and D is sparse then the multiplication is fast.

In all of these we never need to create the full matrices to do the multiplications!

A 1-d wavelet basis of 32 functions



Back to Q and Kriging

$Q = (K\Sigma K^T + R)$ If all the intermediate matrices can be multiplied quickly then so can Q.

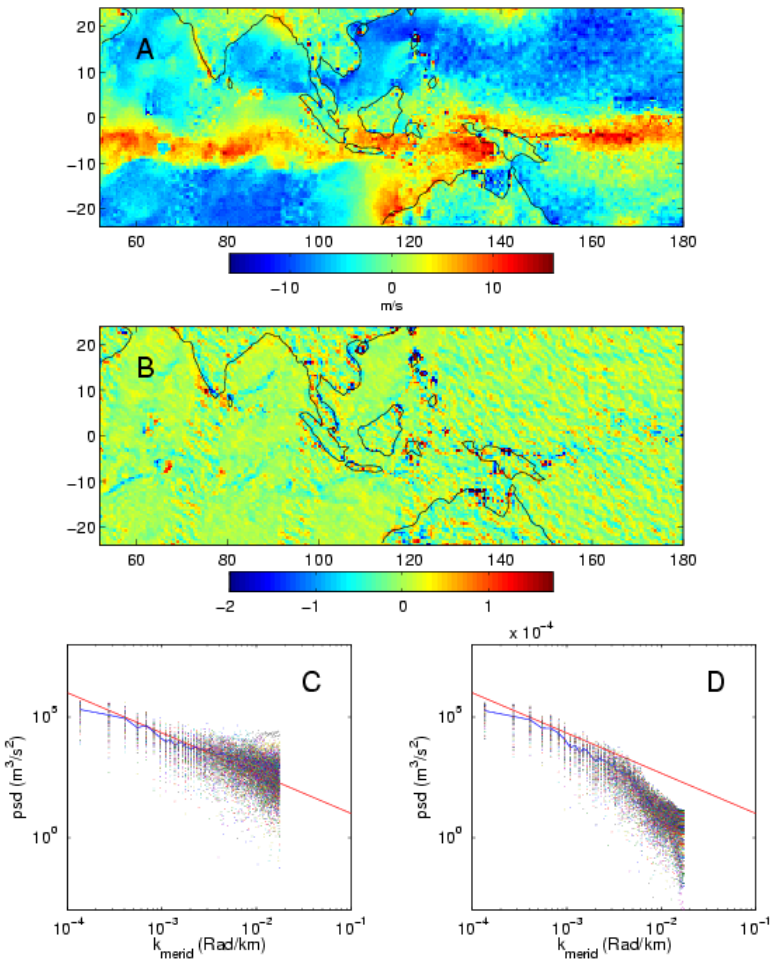
Also note that $\Sigma K\boldsymbol{\omega}$ can also be done quickly.

GibbsWinds

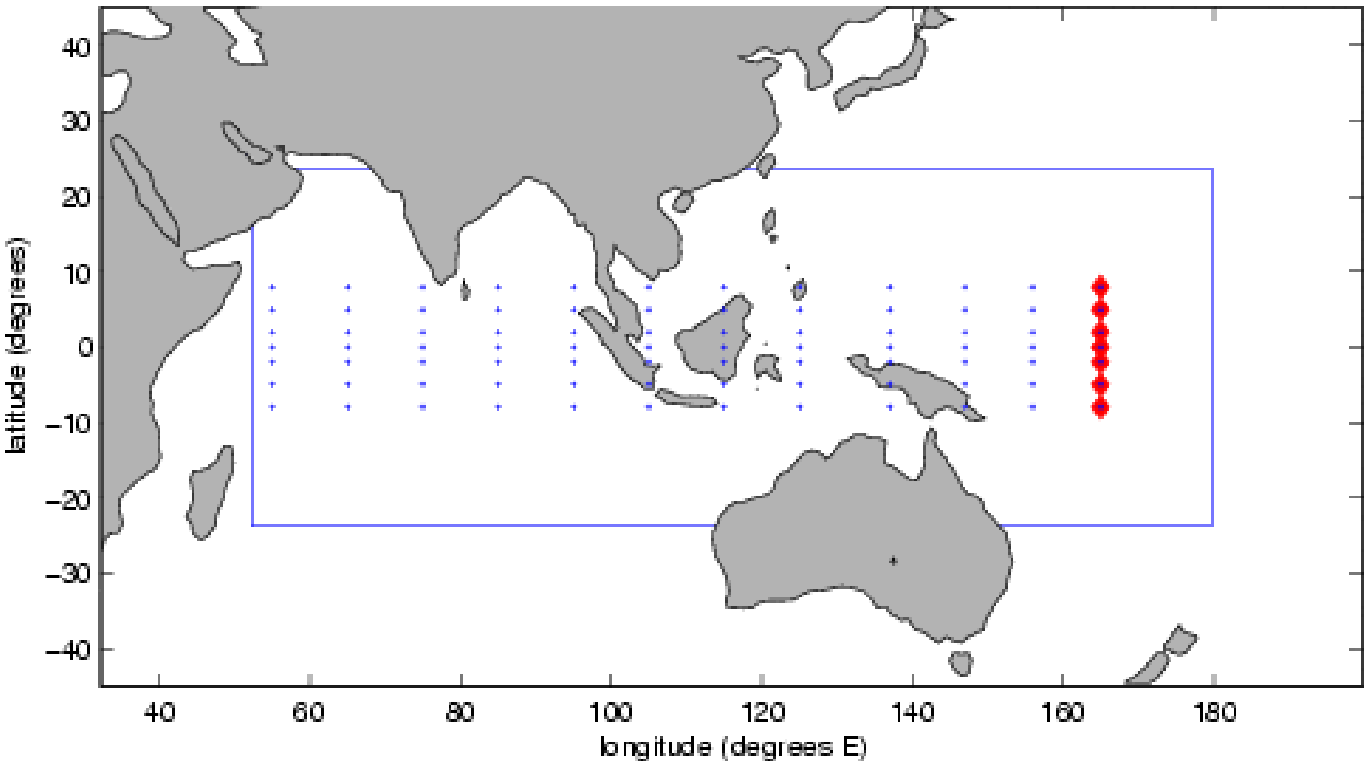
A project to create ocean surface wind fields by blending two forms of data. The CGA was used in the core of a Gibbs sampler using multi-resolution-based covariances.

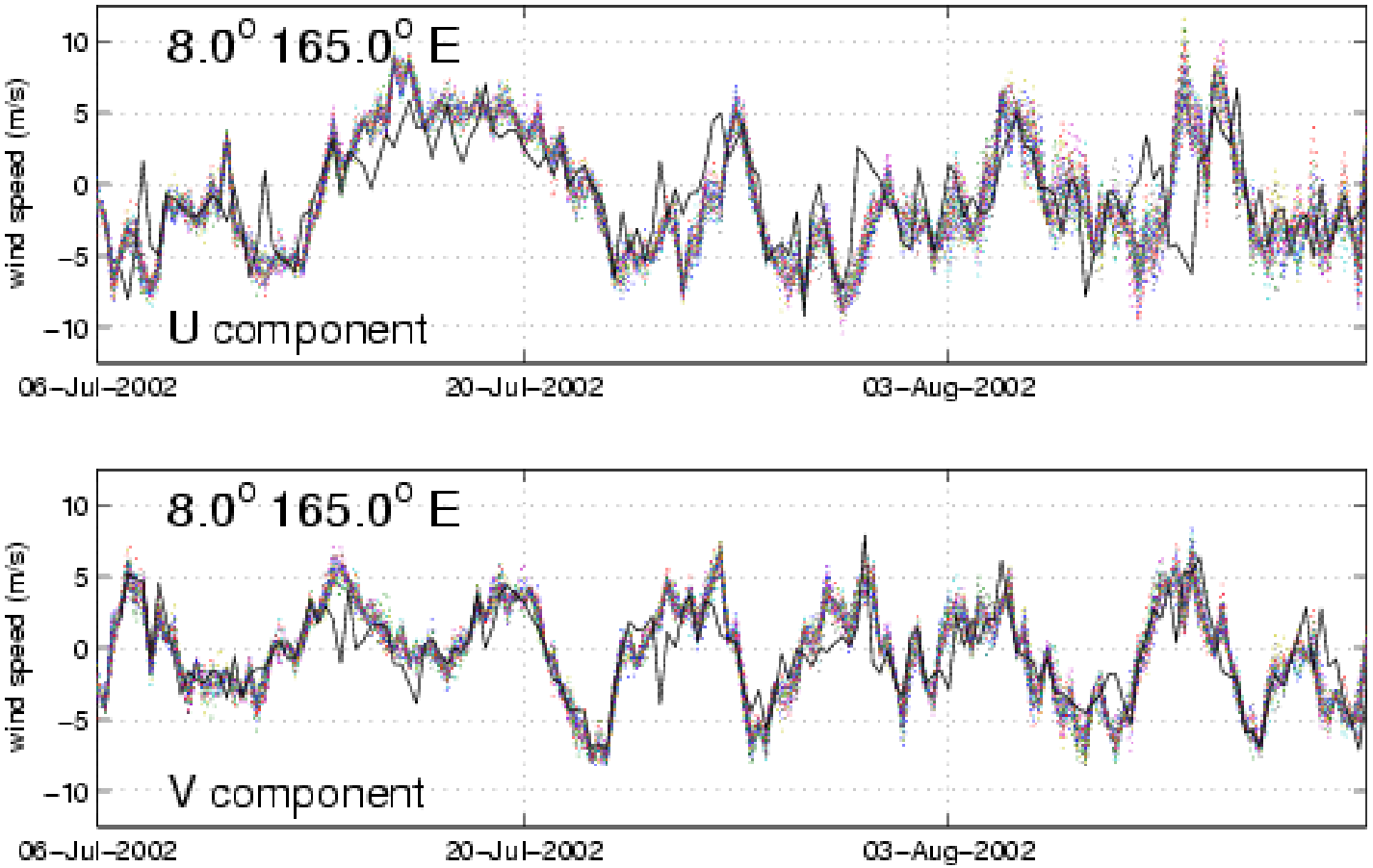
Results are posterior wind realizations for the tropical ocean every 6 hours for three years. 256×96 spatial grid and 4648 time points.

An example of a GibbsWinds realization



Comparison to TAO buoy data




8.0° 165.0° E
U component
wind speed (m/s)
10
5
0
−5
−10
06−Jul−2002
20−Jul−2002
03−Aug−2002
8.0° 165.0° E
V component
wind speed (m/s)
10
5
0
−5
−10
06−Jul−2002
20−Jul−2002
03−Aug−2002

Another approach: Enforcing sparsity

Our job is to find $\boldsymbol{\omega}$ for

$$Q\boldsymbol{\omega} = \boldsymbol{Y}$$

If Q is sparse we can.

- Find a sparse (and exact) Cholesky factorization $Q = CC^T$
- Solve using sparsity $C\boldsymbol{\eta} = \boldsymbol{Y}$
- Solve using sparsity $C^T\boldsymbol{\omega} = \eta$

But how can Q be made sparse?

Tapering Σ

Introduce zeroes by multiplying Σ with a positive definite, compactly supported tapering function, H.

$$\Sigma^*_{i,j} = \Sigma_{i,j} * H_{i,j}$$

(componentwise multiplication)

This certainly introduces zeroes ...

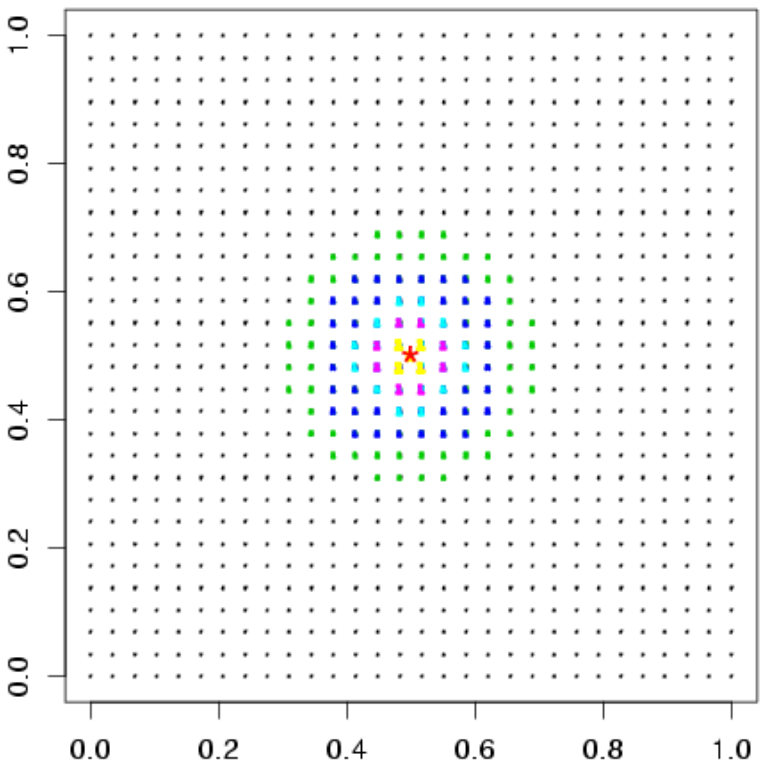
But are we still close to Kriging?

Is it faster?

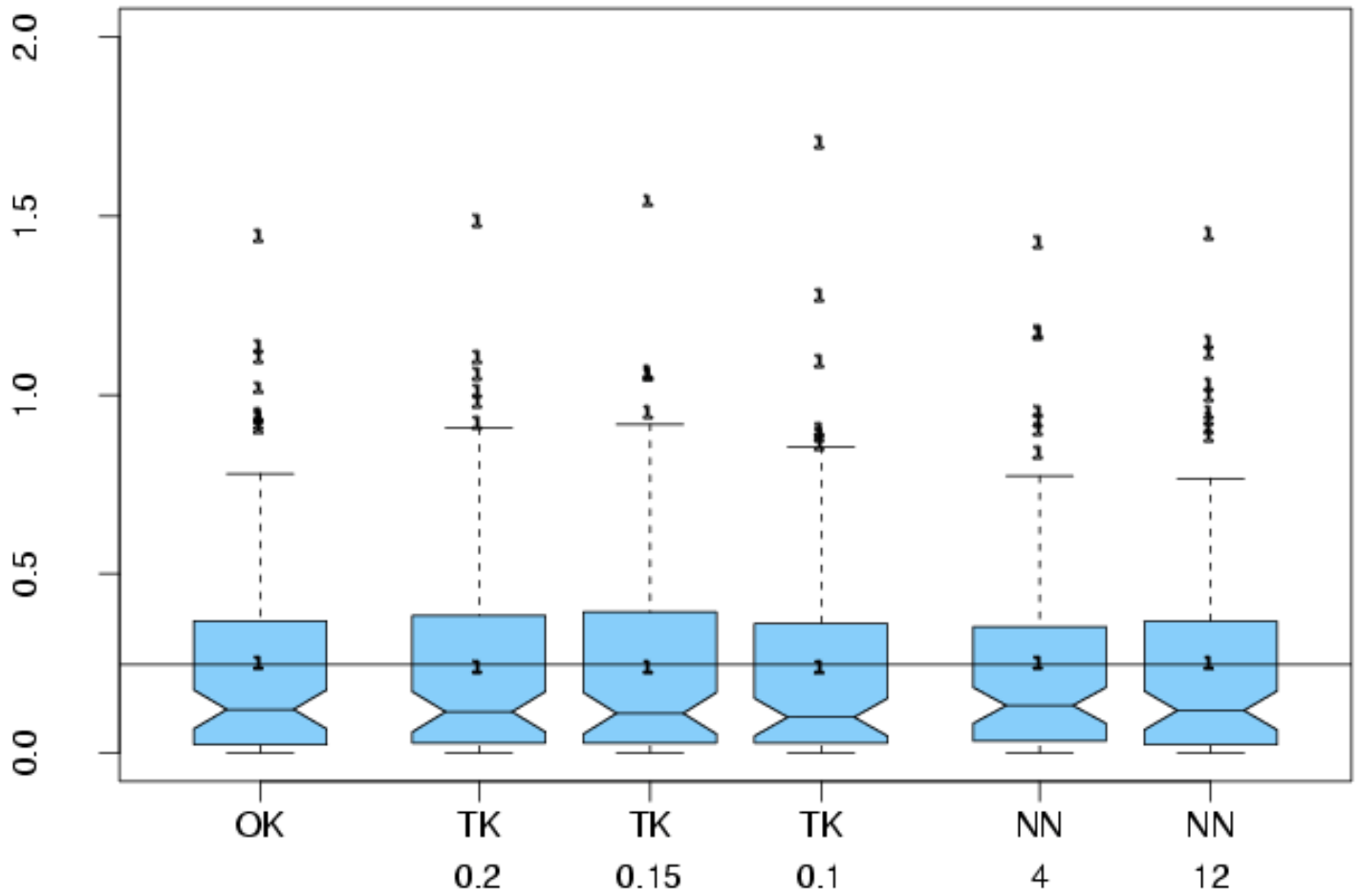
Some numerical results for MSE

30×30 grid predicting at center.

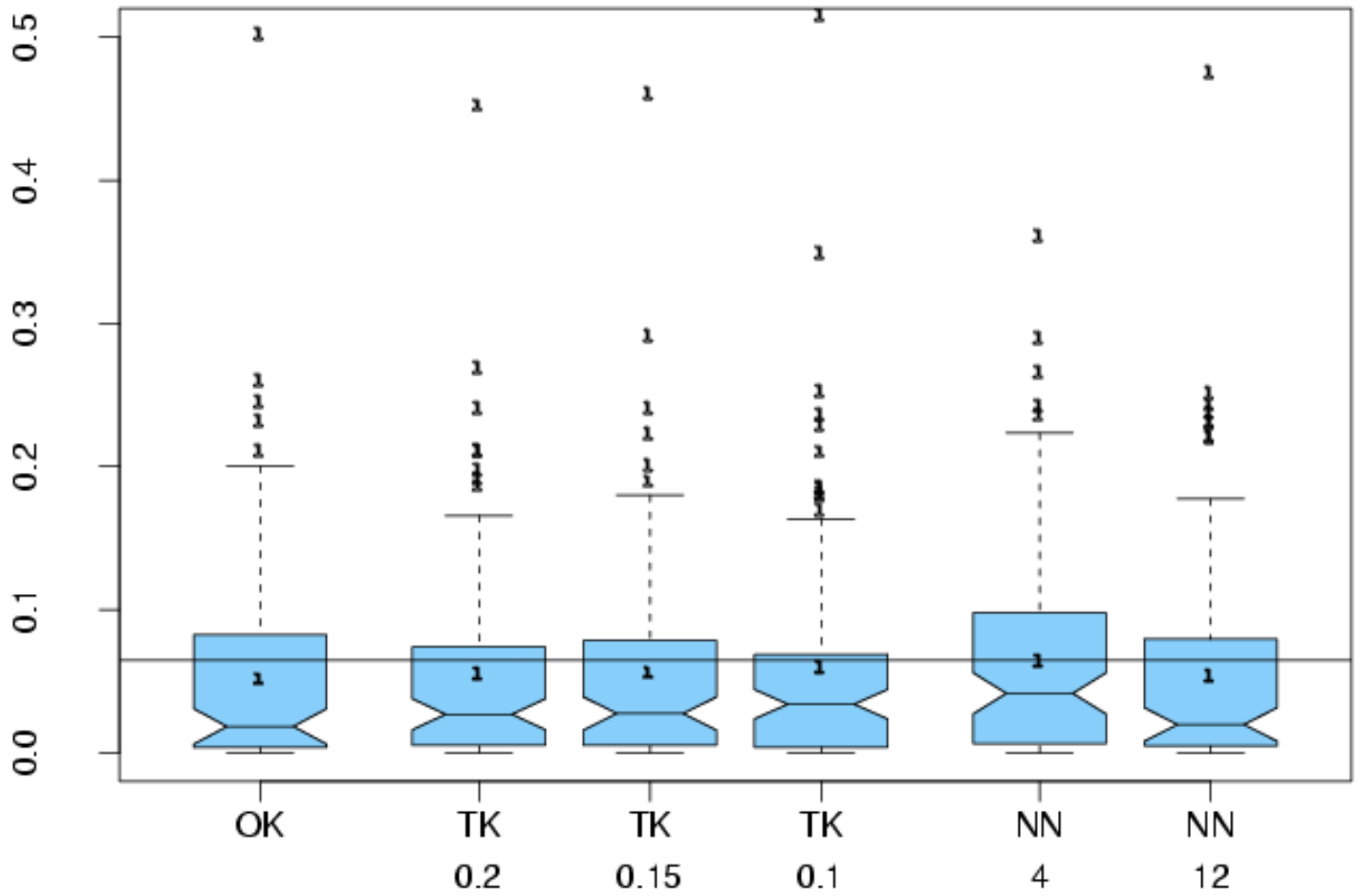
- Ordinary Kriging (OK)
- OK with tapering (.2,.15,.1)
- Nearest Neighbor OK (4,16)



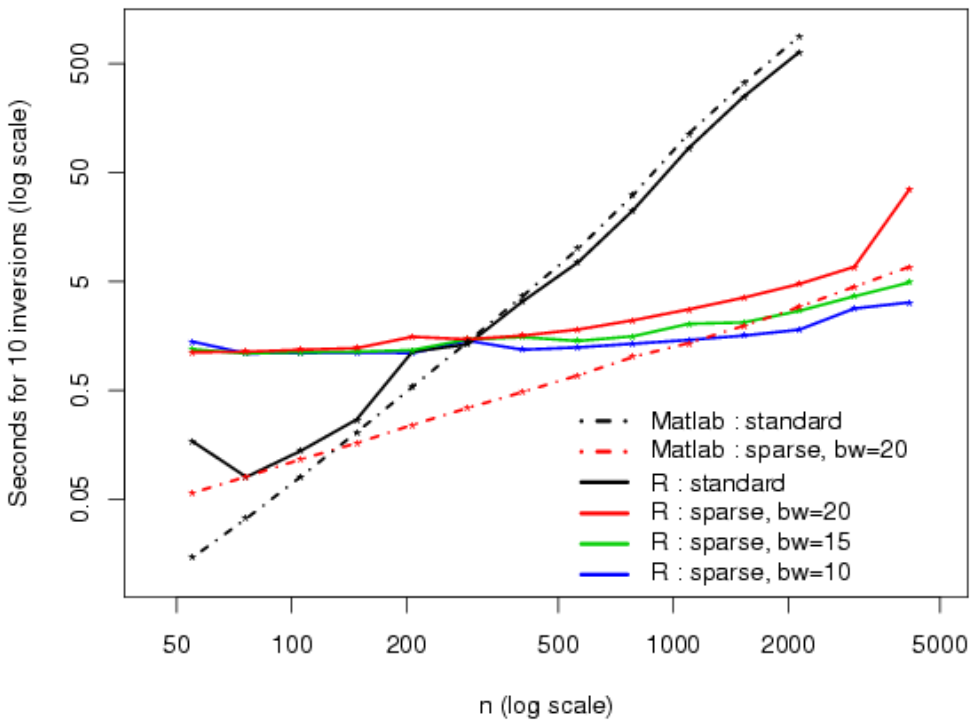
Matern smoothness .75



Matern smoothness 1.25



Some timing results for sparse Kriging



At 2000 points the difference is more than $100:1$**!**

Why does this work?

Working in spectral domain Stationary covariance functions can be uniquely determined by there spectral density (a.k.a. Fourier transform or characterisic function).

From Michael Stein's work if tail of spectral densities are the same then MSE of estimates are asymptotically equivalent.

Effect of tapering:
Convolutions of densities $\longrightarrow$ products of Fourier transforms

Products of densities $\longrightarrow$ convolutions of Fourier transforms

Under certain smoothness on the taper:

$$\lim_{u \to \infty} \frac{\hat{\sigma}(u)}{\int \hat{\sigma}(u-v)\hat{H}(v)dv} = constant$$

Summary

- One can multiply stationary covariance matrices (and multi-resolution matrices) quickly if points are on grids.
- CGA can be used to handle large problems.
- Sparsity can be enforced with little penalty and much improvement in speed. Asymptotic theory of Stein supports these results.

Some open issues:

Companion efficiency in estimating the covariance model.

Relationship to the spatial Kalman filter.