Modellierung dynamischer und räumlicher Prozesse

Edzer J. Pebesma

edzer.pebesma@uni-muenster.de

ifgi
Institute for Geoinformatics
University of Münster

http://ifgi.uni-muenster.de/~epebe_01/

16 October 2007

Course overview – grading, presence etc.

- Vorlesung: Edzer Pebesma, Übungen: Kristina Helle; Katharina Henneböhl.
- Organization: see web site.

Course overview – Literature

- C. Chatfield, The analysis of time series: an introduction. Chapman and Hall: chapters 1, 2 and 3
- Applied Spatial Data Analysis with R, by R. Bivand, E. Pebesma and V. Gomez-Rubio (Springer;):
  - Ch 1, 2, 3
  - Ch 4, 5, 6 (whatever is convenient from it)
  - **Ch 8 (geostatistics)**

Domain – interest poll

Spatio-temporal modelling is a large subject.

- geology/geophysics
- hydrology
  - ground water dynamics
  - rainfall-runoff modelling
- ecology
  - plant/species dynamics
  - species associations, biodiversity
  - cause-effects, dose-response
- paleo time scales (climate records)
- natural hazards
- people:
  - spatial planning
  - demography
  - environmental health assessment (air quality)

Point of departure – where are we?

What is

- correlation, covariance
- t-test
- linear regression, ANOVA
- 95% confidence interval
- spatial correlation, temporal correlation
- time domain vs. frequency domain modelling
- kriging
- least squares solution, normal equations
- a partial differential equation
- a Kalman filter

Aims of modelling

... could be

- it is good fun
- studying models is easier than studying the world around us
- they live mostly in computers
- there are so many different models
- models really *use* my CPU

Scientific aims of modelling are

- to learn about the world around us
- to predict the past, current or future, in case where measurement is not feasible.

Aims of modelling

... could be

- it is good fun
- studying models is easier than studying the world around us
- they live mostly in computers
- there are so many different models
- models really *use* my CPU

Scientific aims of modelling are

- to learn about the world around us
- to predict the past, current or future, in case where measurement is not feasible.

What is a model?

- conceptual models
- object models (e.g., UML)
- mathematical models

What is a mathematical model?

A mathematical model is an abstract model that uses mathematical language to describe the behaviour of a system.

a representation of the essential aspects of an existing system (or a system to be constructed) which presents knowledge of that system in usable form (P. Eykhoff, 1974, System Identification, J. Wiley, London.)

In the natural sciences, a model is always an approximation, a simplification of reality. If degree of approximation meets the required accuracy, the model is useful, or valid (of value). A validated model does not imply that the model is "true"; more than one model can be valid at the same time.

What is a mathematical model?

A mathematical model is an abstract model that uses mathematical language to describe the behaviour of a system. *a representation of the essential aspects of an existing system (or a system to be constructed) which presents knowledge of that system in usable form* (P. Eykhoff, 1974, System Identification, J. Wiley, London.)

In the natural sciences, a model is always an approximation, a simplification of reality. If degree of approximation meets the required accuracy, the model is useful, or valid (of value). A validated model does not imply that the model is “true”; more than one model can be valid at the same time.

What is a mathematical model?

A mathematical model is an abstract model that uses mathematical language to describe the behaviour of a system.
a representation of the essential aspects of an existing system (or a system to be constructed) which presents knowledge of that system in usable form (P. Eykhoff, 1974, System Identification, J. Wiley, London.)
In the natural sciences, a model is always an approximation, a simplification of reality. If degree of approximation meets the required accuracy, the model is useful, or valid (of value). A validated model does not imply that the model is “true”; more than one model can be valid at the same time.

What may follow?

(In no particular order)

- simple models, e.g. cascade models
- models that have coupled equations; differential equations
- models built on random processes, e.g. Gaussian dispersion models
- modelling data: time series modelling, models using spatial correlation
- model bias, and spatio-temporal bias correction
- finding model parameters: model calibration, inverse modelling
- combining models and measurements: kalman filtering

(not necessarily in this order)

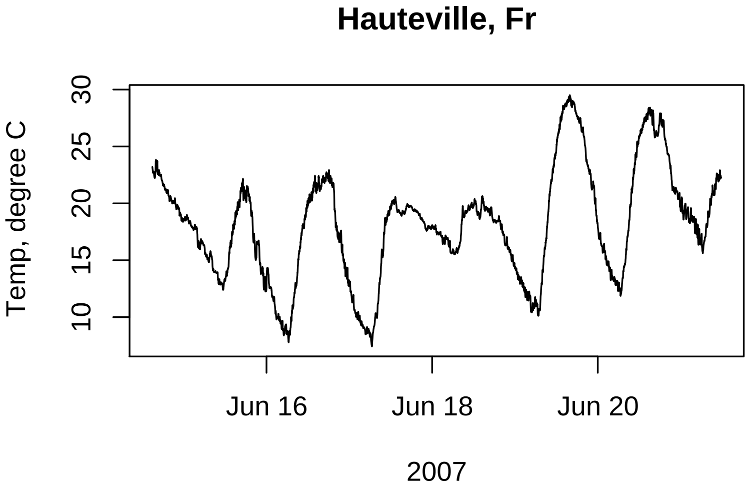
Time series models

We will first look into time series models, because they are

- simple
- easy to write down
- well understood

Time series models are roughly divided in (a) time domain models and (b) frequency domain models. Time domain models look at correlations and memory, frequency domain concentrate on periodicities. Spatial equivalents are mostly found in (a), although (b) has spatial equivalences as well (e.g. wavelets).

Consider the following process ($\Delta t = 1$ min):



Questions

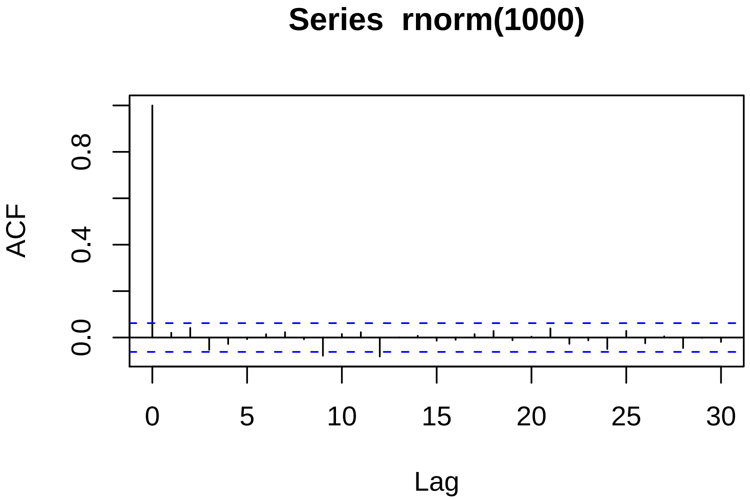
- how can we describe this process in statistical terms?
- how can we model this process?
- (how) can we predict future observations?

White noise, AR(n)

Perhaps the simplest ts model is white noise with mean m:

$$y_t = m + e_t, \quad e_t \sim N(0, \sigma^2)$$

$N(0, \sigma^2)$ denoting the normal distribution with mean 0 and variance σ^2, and $\sim$ meaning "distributed as" or "coming from". A white noise process is completely without memory: each observation is independent from its past or future. We can look at the auto-correlation function of a white noise process, and find it is uncorrelated:


Series rnorm(1000)
ACF
0.0
0.4
0.8
0
5
10
15
20
25
30
Lag

Autocorrelation

Autocorrelation (or lagged correlation) is the correlation between y_i and y_{i+h}, with h the lag:

$$r(h) = \frac{\sum_{i=1}^{n-h}(y_i - \bar{y})(y_{i+h} - \bar{y})}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

with $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$

Random walk

A simple, next model to look at is that of *Random walk*, where each time step a change is made according to a white noise process:

$$y_t = y_{t-1} + e_t$$

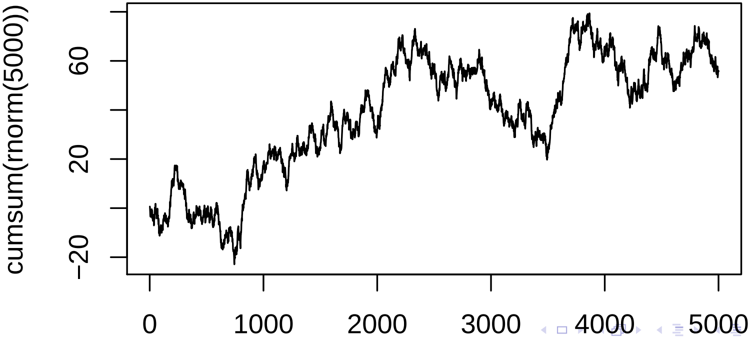
Such a process has memory, and long-range correlation. If we take the first-order differences,

$$y_t - y_{t-1} = e_t$$

we obtain the white noise process.
Further, the variance of the process increases with increasing domain (i.e., it is non-stationary)

Example random walk:

We can compute it as the cumulative sum of standard normal deviates: $y_n = \sum_{i=1}^{n} e_i$:



MA(1), MA(q)

Let e_t be a white noise process. A moving average process of order q is generated by

$$y_t = \beta_0 e_t + \beta_1 e_{t-1} + ... + \beta_q e_{t-q}$$

Note that the β_j are weights, and could be $\frac{1}{q+1}$ to obtain an unweight average. Moving averaging smoothes the white noise series e_t.

AR(1), AR(p)

An auto-regressive (1) model, or AR(1) model is generated by

$$y_t = \phi_1 y_{t-1} + e_t$$

and is sometimes called a Markov process. Given knowledge of y_{t-1}, observations further back carry no information; more formally:

$$\Pr(y_t | y_{t-1}, y_{t-2}, ..., y_{t-q}) = \Pr(y_t | y_{t-1})$$

- $\phi_1 = 1$ gives random walk, $\phi_1 = 0$ gives white noise.
- AR(1) have correlations beyond lag 1
- AR(1) have "zero" *partial* autocorrelations beyond lag 1

AR(p)

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + ... + \phi_p y_{t-p} + e_t$$

or

$$y_t = \sum_{j=1}^{p} \phi_j y_{t-j} + e_t$$

- The state of y_t does not *only* depend on y_{t-1}, but observations further back contain information
- AR(p) have autocorrelations beyond lag p
- AR(p) have "zero" *partial* autocorrelations beyond lag p

Next: what is partial correlation?

What is partial correlation?

- Correlation between y_t and y_{t-2} is simply obtained by plotting both series of length $n-2$, and computing correlation
- Lag-2 *partial* autocorrelation of y_t and y_{t-2}, given the value inbetween y_{t-1} is obtained by
  - computing residuals $\hat{e}_t$ from regressing of y_t on y_{t-1}
  - computing residuals $\hat{e}_{t-2}$ from regressing of y_{t-2} on y_{t-1}
  - computing the correlation between both residual series $\hat{e}_t$ and $\hat{e}_{t-2}$.
- Lag-3 partial autocorrelation regresses y_t and y_{t-3} on *both* intermediate values y_{t-1} and y_{t-2}
- etc.

Partial correlation can help reveal what the order of an AR(p) series is.

relation between AR and MA processes

substitute the AR(1) as follows

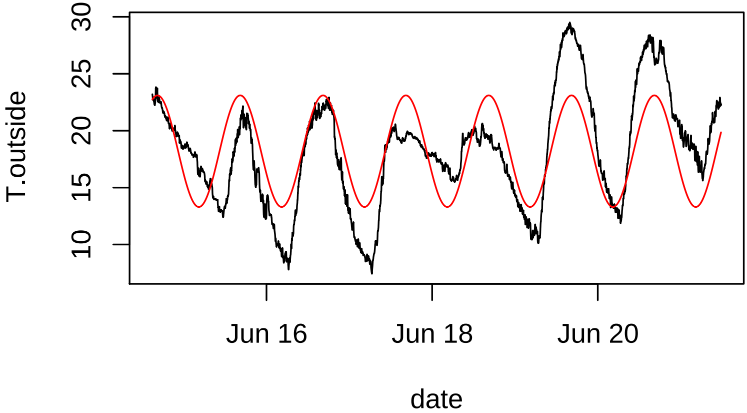
$$y_t = \phi_1 y_{t-1} + e_t$$

$$y_t = \phi_1(\phi_1 y_{t-2} + e_{t-1}) + e_t$$

$$y_t = \phi_1^2(\phi_1 y_{t-3} + e_{t-2}) + \phi_1 e_{t-1} + e_t$$

etc. In the limit, we may write any AR process as an (infinite) MA process, and vice versa.

Back to the temperature series



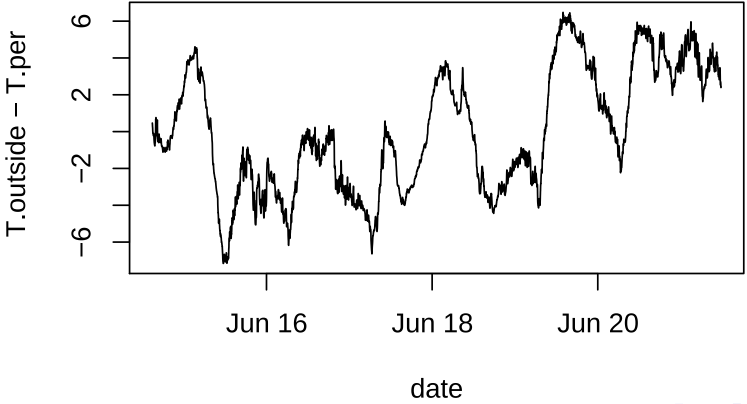
Removing the diurnal periodicity

Assuming this is a sinus function, $\alpha_1 + \alpha_2 \sin(t + \alpha_3)$, we need non-linear regression (α_3)

```
> f = function(x) sum((T.outside -
        (x[1]+x[2]*sin(pi*(hours+x[3])/12)))^2)
> nlm(f,c(0,0,0))
$minimum
[1] 108956.1

$estimate
[1] 18.189544 -4.904740  1.604442
...
> T.per = 18.2-4.9*sin(pi*(hours+1.6)/12)
> plot(T.outside,type=’l’)
> plot(T.per, type=’l’, col = ’red’, add=TRUE)
```

Temperature anomaly



What can we do with such models?

- try to find out which model fits best (model selection)
- learn how they were/could have been generated
- predict future observations (estimation/prediction/forecasting)
- generate similar data ourselves (simulation)

How to select a “best” model?

A possible approach is to find the minimum for *Akaike’s Information Criterion* (AIC) for ARMA(p,q) models and series of length n:

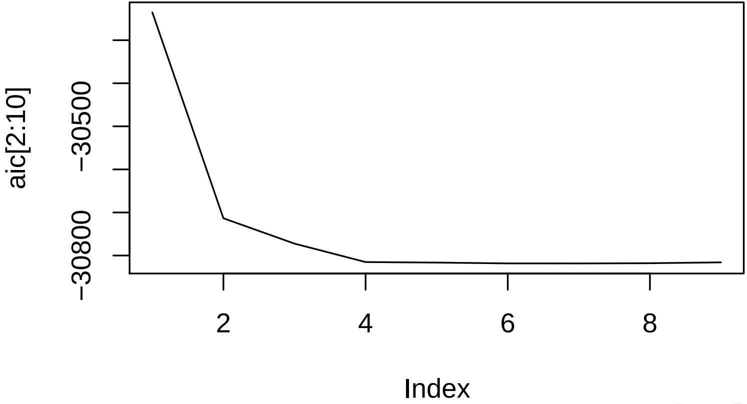
$$AIC = \log \hat{\sigma}^2 + 2(p+q+1)/n$$

with $\hat{\sigma}^2$ the estimated residual (noise) variance.
Instead of finding a single best model using this single criterion, it may be better is to select a small group of “best” models, and look at model diagnostics for each: is the residual white noise? does it have stationary variance?
Even better may be to keep a number of “fit” models and consider each as (equally?) suitable candidates.

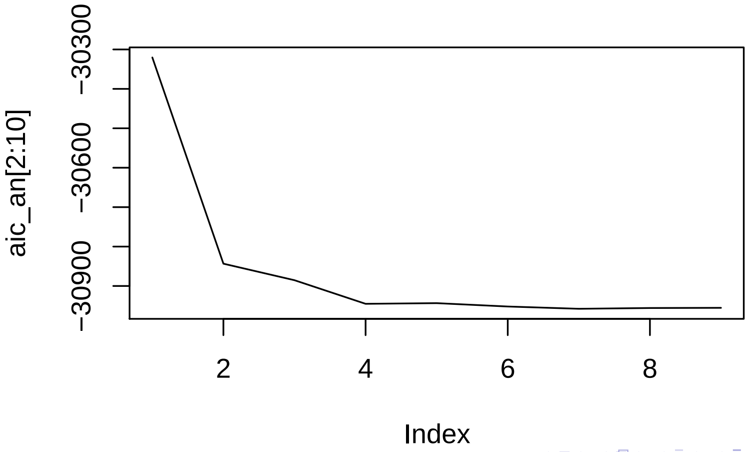
AIC for AR(p)

```
> arima(temp,c(1,0,0))$aic  # AR(1)
[1] -23547.93
> arima(temp,c(2,0,0))$aic  # AR(2)
[1] -30235.42
> arima(temp,c(3,0,0))$aic  # etc.
[1] -30713.51
> arima(temp,c(4,0,0))$aic
[1] -30772.31
> arima(temp,c(5,0,0))$aic
[1] -30815.14
> arima(temp,c(6,0,0))$aic
[1] -30816.35
> arima(temp,c(7,0,0))$aic
[1] -30818.27
> arima(temp,c(8,0,0))$aic
[1] -30818.39
> arima(temp,c(9,0,0))$aic
[1] -30817.82
> arima(temp,c(10,0,0))$aic
[1] -30815.84
```

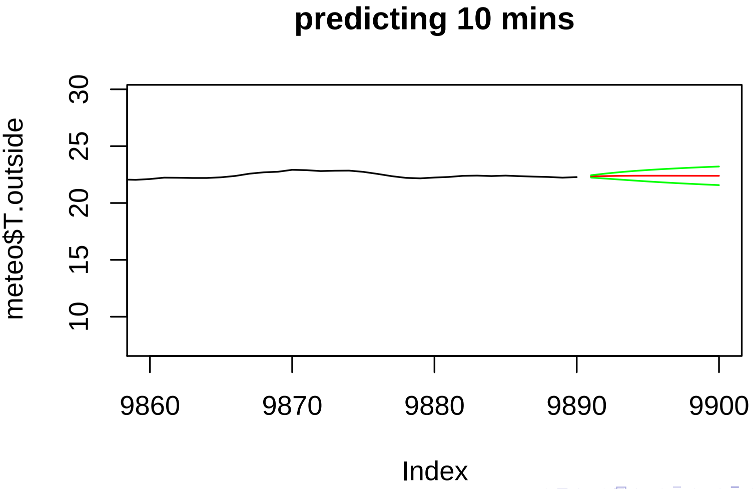
AIC as a function of p, for AR(p)



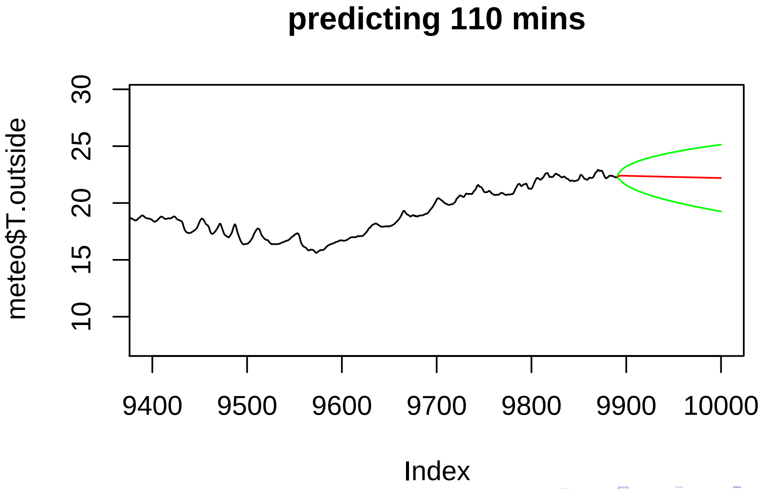
Anomaly AIC as a function of p, for AR(p)



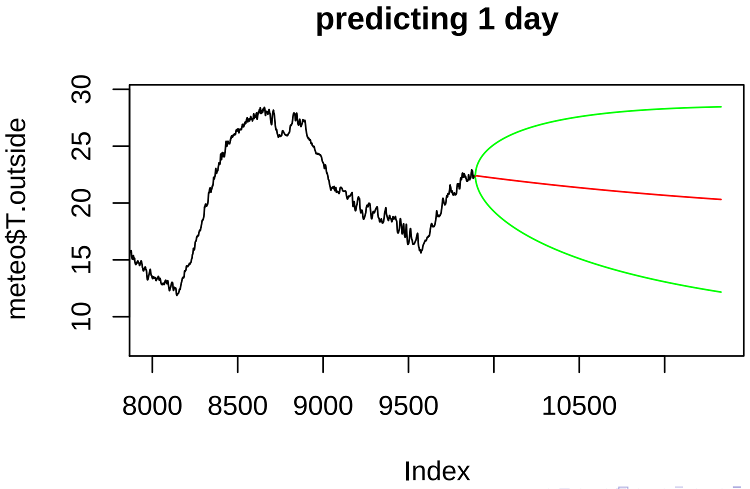
Prediction AR(6): 10 minutes



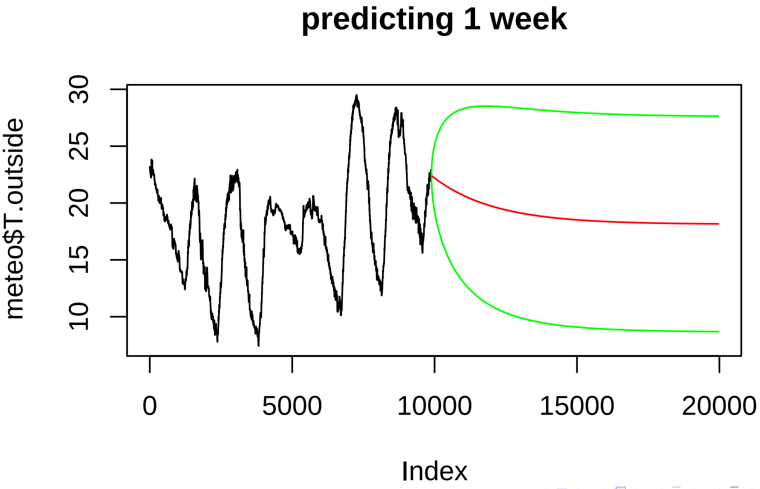
Prediction AR(6): 2 hours



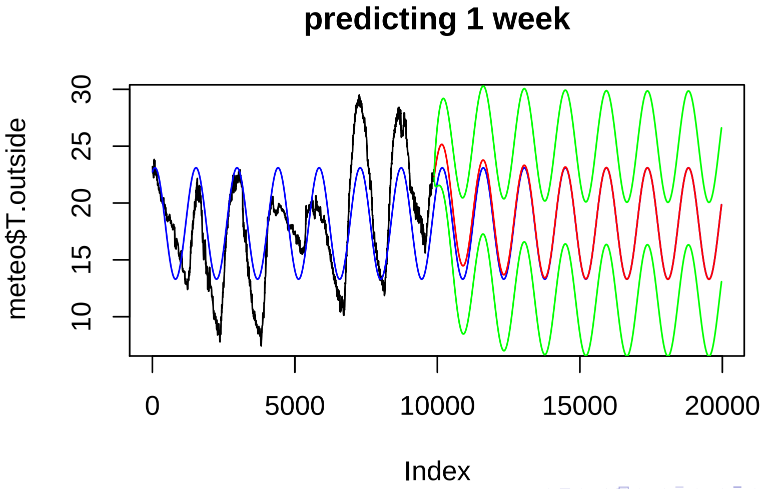
Prediction AR(6): 1 day



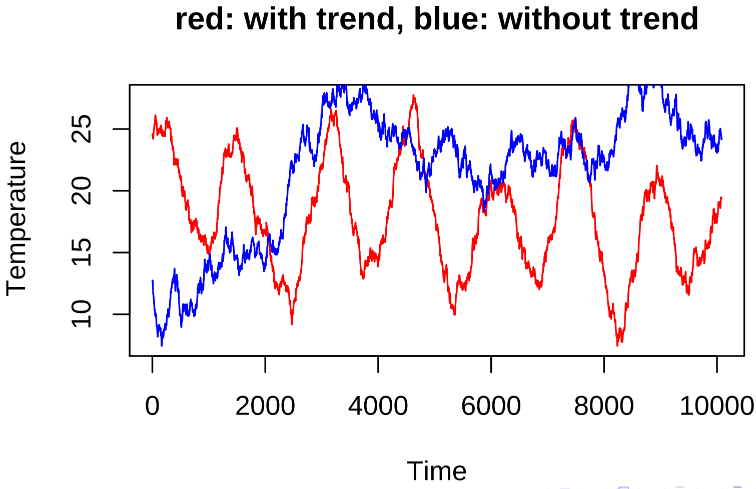
Prediction AR(6): 1 week



Prediction sinus trend + AR(6): 1 week



Simulation: with and without trend



What can we learn from this?

Prediction/forecasting:

- AR(6) prediction is a compromise between the end of the series and the trend
- the closer we are to observations, the more similar the prediction is to the nearest (last) observation
- further in the future the prediction converges to the trend
- the more useful (realistic) the trend is, the more realistic the far-into-the-future prediction becomes
- the standard error of prediction increases when predictions are further in the future.

Optimization: 1. linear systems

Take the example

$$\begin{aligned} ax_{11} + bx_{12} &= y_1 \\ ax_{21} + bx_{22} &= y_2 \end{aligned}$$

with the x and y values known, and a and b unknown. This is similar to fitting a straight line through two points: let (x_1, y_1) be the first point and (x_2, y_2) be the second, then

$$\begin{aligned} a + bx_1 &= y_1 \\ a + bx_2 &= y_2 \end{aligned}$$

The approach is substition: rewrite one equations such that isolates a or b, and substitute that in the second.

Matrix notation

We can rewrite

$$\begin{aligned} ax_{11} + bx_{12} &= y_1 \\ ax_{21} + bx_{22} &= y_2 \end{aligned}$$

as the matrix product

$$\left[\begin{array}{cc} x_{11} & x_{12} \\ x_{21} & x_{22} \end{array}\right]\left[\begin{array}{c} a \\ b \end{array}\right] = \left[\begin{array}{c} y_1 \\ y_2 \end{array}\right]$$

or

$$Xa = y$$

Matrix transposition

The transpose of a matrix is the matrix formed when rows and columns are reversed. If

$$A = \begin{bmatrix} 1 & 4 \\ 2 & -1 \\ 8 & 9 \end{bmatrix}$$

then it's transpose,

$$A' = \begin{bmatrix} 1 & 2 & 8 \\ 4 & -1 & 9 \end{bmatrix}$$

(and may be written as A^T)

Matrix inverse and identity

The identity matrix is square (nr of rows equals nr of columns), has ones on the diagona (for which the row number equals the column number) and zeroes elsewhere. E.g. the 3×3 identity

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The *inverse* of a square matrix X, X^{-1}, is *defined* by the products

$$X^{-1}X = I$$

and

$$XX^{-1} = I$$

Suppose we have n equations with p unknowns:

$$\begin{array}{ccccccc} a_1x_{11}+a_2x_{12}+ & \ldots & + & a_px_{1p} & = & y_1 \\ a_1x_{21}+a_2x_{22}+ & \ldots & + & a_px_{2p} & = & y_2 \\ \vdots & \ddots & & \vdots & & \vdots \\ a_1x_{n1}+a_2x_{n2}+ & \ldots & + & a_px_{np} & = & y_n \end{array}$$

we can rewrite this in matrix notation as $Xa = y$, with x_{ij} corresponding to element (i, j) (row i, column j) in X, having n rows and p columns; a and y column vectors having p and n elements, respectively. Now, X and y are known, and a is unknown. a Solutions:

- if $p > n$, there is no single solution
- if $p = n$ and X is not singular, then $a = X^{-1}y$
- if $p < n$ we have an overdetermined system, and may e.g. look for a least square (best approximating) solution.

Linear least squares solution

If $p < n$, a solution usually does not exist: try fitting a straight line through three or more arbitrary points.

Now rewrite $Xa = y$ as $y = Xb + e$, with e the distance (in y-direction) from the line. If we want to minimize the sum of squared distances, then we need to find b for which $R = \sum_{i=1}^{n} e_i^2$ is minimum. In matrix terms, $R = (y - Xb)'(y - Xb)$ with $'$ denoting transpose (row/col swap).

$$\frac{\delta R}{\delta b} = 0$$

$$\frac{\delta (y - Xb)'(y - Xb)}{\delta b} = 0$$

$$\frac{\delta (y'y - (Xb)'y - y'(Xb) + (Xb)'Xb)}{\delta b} = 0$$

$$\frac{\delta(y'y - (Xb)'y - y'(Xb) + (Xb)'Xb)}{\delta b} = 0$$

now you should first note that $(Xb)' = b'X'$, and second that $b'X'y = y'Xb$ because these are scalars. Then,

$$-2X'y + 2X'Xb = 0$$

$$X'Xb = X'y$$

$$b = (X'X)^{-1}X'y$$

this yields the least squares solution for b; the solution equations are called the *normal equations*.

The practice of solving systems

when we write

$$Ax = b$$

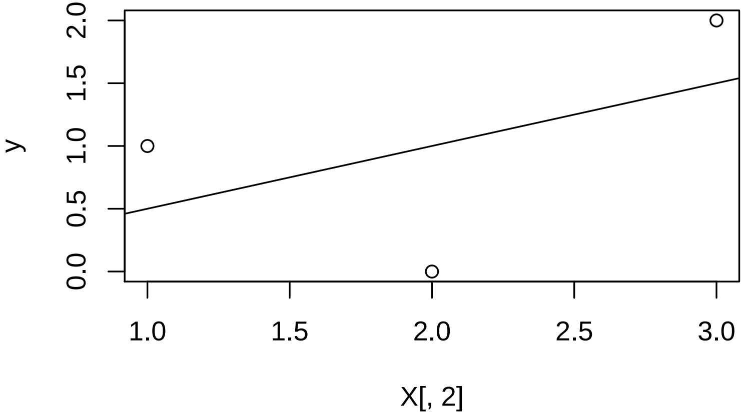
with known A and b and unknown x, the solution is

$$x = A^{-1}b$$

In practice however, we do not need to compute A^{-1}, but can directly solve for x. This is much cheeper.

```
> m=matrix(0,3000,3000)
> diag(m)=1
> system.time(x <- solve(m))
   user  system elapsed
   1.52    0.56    2.08
> system.time(x <- solve(m,rep(0,3000)))
   user  system elapsed
  0.396   0.148   0.545
```

```
> X=cbind(c(1,1,1),c(1,2,3))
> X
     [,1] [,2]
[1,]    1    1
[2,]    1    2
[3,]    1    3
> y = c(1,0,2)
> solve(t(X) %*% X, t(X) %*% y)
     [,1]
[1,]  0.0
[2,]  0.5
```



y
0.0
0.5
1.0
1.5
2.0
1.0
1.5
2.0
2.5
3.0
X[, 2]

Non-linear Optimization

- one-dimensional search on a unimodal function: golden search
- non-linear least squares: the Gauss Newton algorithm
- probabilistic methods: global search
  - Metropolis-Hastings
  - Simulated Annealing

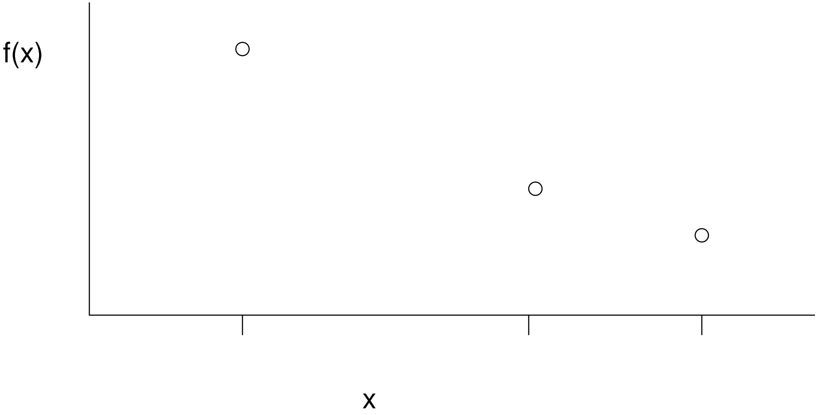
Golden search

Golden ratio:

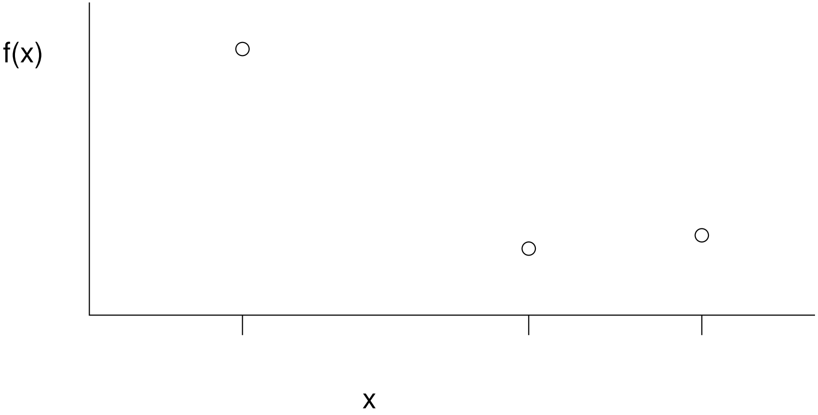
$$\frac{x_1}{x_2} = \frac{x_2}{x_1 + x_2}$$

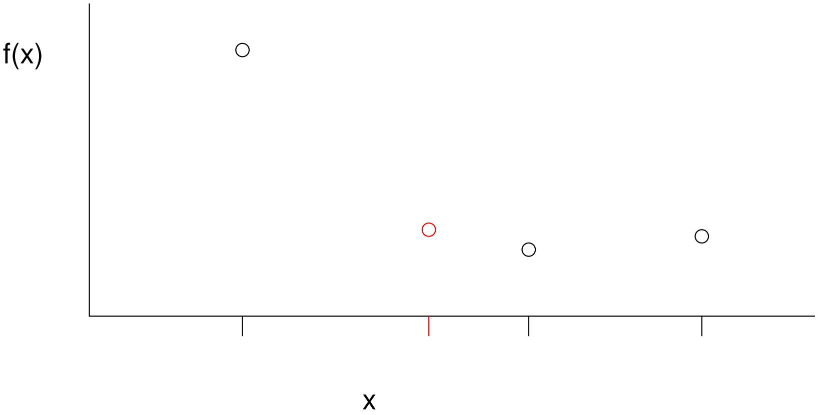
Solution (check): if $x_1 = 1$, then $x_2 \approx 1.618$ or $x_2 \approx 0.618$
Found in: art, sculpture, geometry (pentagrams), Egyptian pyramides, architecture, nature, A4 paper, ...

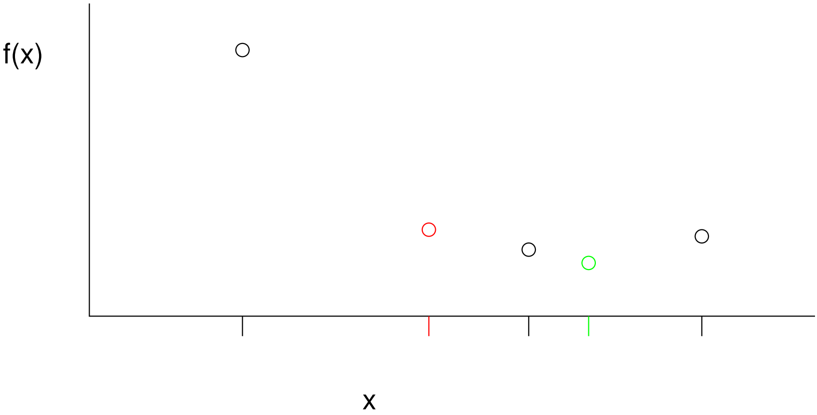
Minimum outside current section

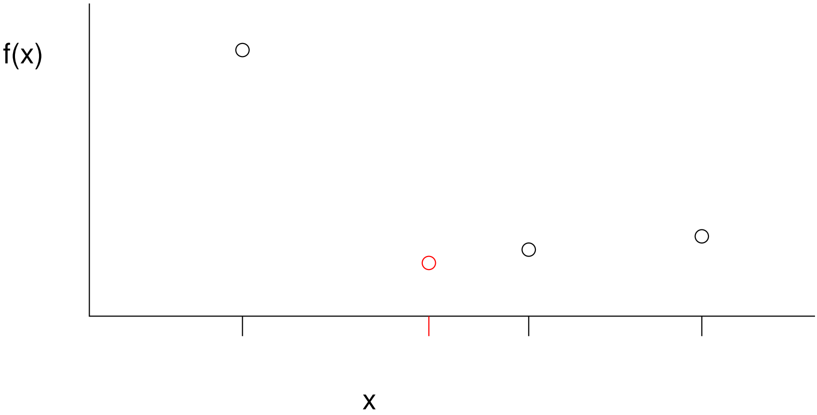


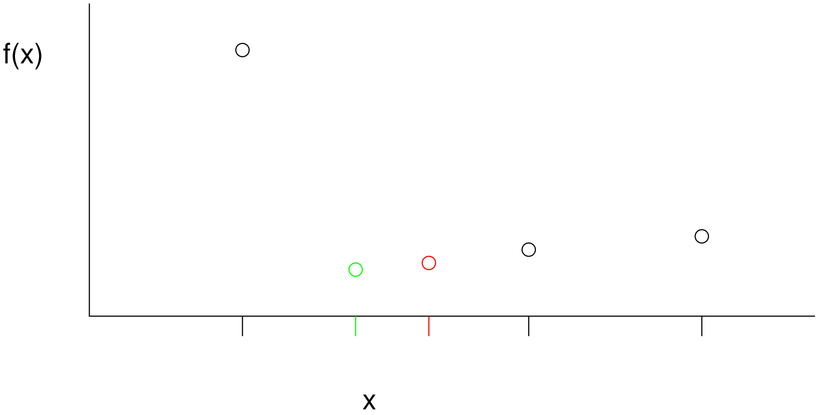
Minimum inside current section




f(x)
x


f(x)
x


f(x)
x

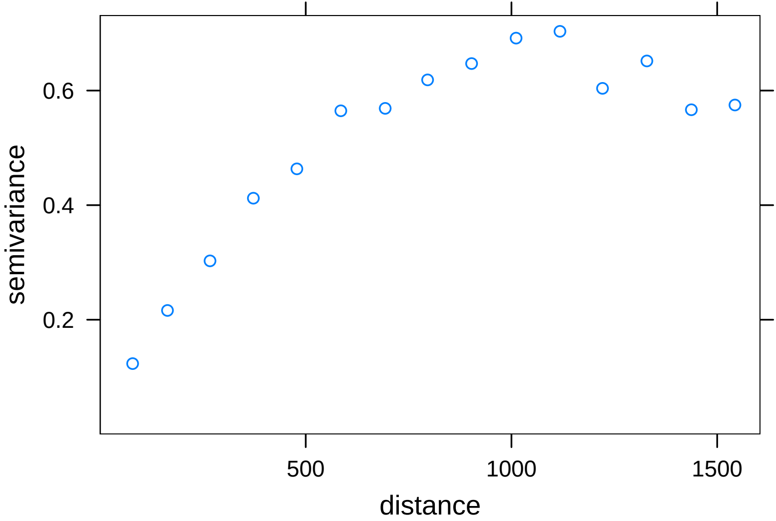

f(x)
x

Algorithm

Recursive zooming:

1. find three GR points, a, b and c such that the minimum lies within a and b
2. put a point d in the largest section according to GR, with the smallest interval closest to the smallest value
3. (In case of adbc) determine whether the mininum is between a and b or d and c
4. continue with either adb or dbc as if it were abc, unless we're sufficiently close*

* in terms of our goal, or of numerical resolution


semivariance
0.2
0.4
0.6
500
1000
1500
distance

Combined linear and golden search

Spherical variogram with nugget has three parameters: nugget c_0, (partial) sill c_1 and range a:

$$\gamma(h) = \begin{cases} 0 & \text{if} \quad h = 0 \\ c_0 + c_1 f(a,h) & \text{if} \quad h > 0 \end{cases}$$

with

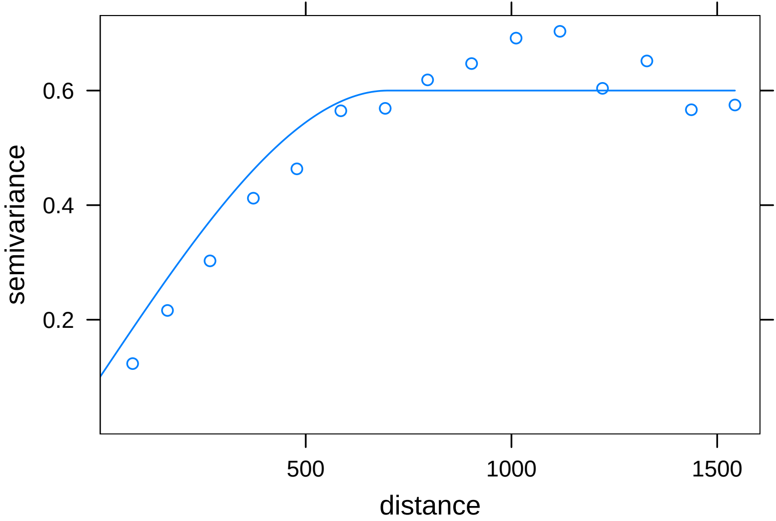
$$f(a,h) = \begin{cases} \frac{3h}{2a} - \frac{1}{2}(\frac{h}{a})^3 & \text{if} \quad 0 \leq h \leq a \\ 1 & \text{if} \quad h > a \end{cases}$$

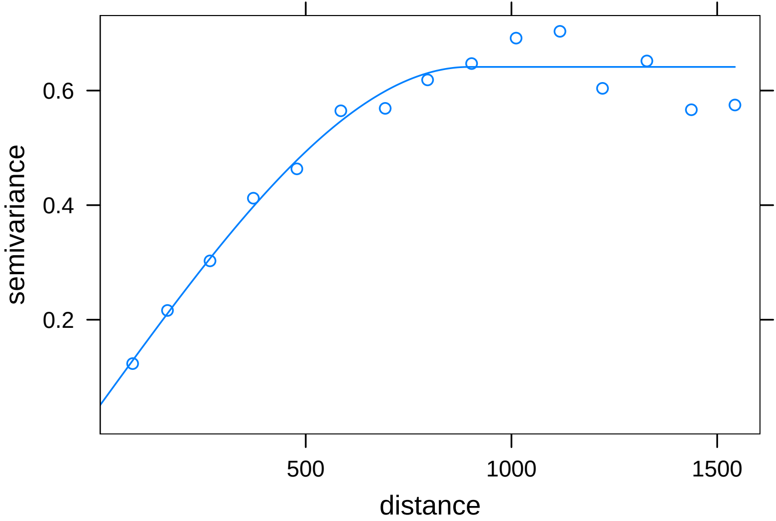
Approach:

Provide an initial estimate a_0; then iterate:

1. given current fit for a, fit the linear coefficients c_0 and c_1
2. given this fit, do golden search for a

until convergence (vector (a, c_0, c_1) does not move).


semivariance
0.2
0.4
0.6
500
1000
1500
distance


semivariance
0.2
0.4
0.6
500
1000
1500
distance

Gauss-Newton

Golden search may be used for any criterion, e.g. $f(x) = \sum_{i=1}^{n} g_i(x)^p$ for any chosen p. If we limit ourselves to *least squares* (i.e., $p = 2$) and want to generalize this for higher dimensional (i.e., multiple parameter) x (e.g. $x = [x_1, ..., x_q]'$) we may use the Gauss-Newton algorithm (non-linear least squares).

Gauss-Newton: the algorithm (1/2)

Problem: given a model $y = g(X, \theta) + e$ find

$$\min_\theta \sum (y - g(X, \theta))^2$$

Let $f_i(\theta) = y_i - g(X_i, \theta)$, so we minimize $R = \sum_{i=1}^{n} (f_i(\theta))^2$
This is a problem from space $(1 \times n)$ to $(1 \times m)$
Given a starting value θ^0 we search the direction of steepest descent in terms of R, using first order derivatives of R towards θ.
By iteration, from θ^k we find θ^{k+1} by

$$\theta^{k+1} = \theta^k + \delta^k$$

until we have convergence.

Gauss-Newton algorithm (2/2)

Let the Jakobian be

$$J_f(\theta^k) = \begin{bmatrix} \frac{\delta f_1(\theta^k)}{\delta\theta_1} & \cdots & \frac{\delta f_1(\theta^k)}{\delta\theta_m} \\ \vdots & \ddots & \vdots \\ \frac{\delta f_n(\theta^k)}{\delta\theta_1} & \cdots & \frac{\delta f_n(\theta^k)}{\delta\theta_m} \end{bmatrix}$$

In

$$\theta^{p+1} = \theta^p + \delta^k$$

we find δ^k by solving

$$J_f(\theta_k)' J_f(\theta_k)\delta^k = -J_f(\theta_k)' f(\theta^k)$$

What if $\delta f_n(\theta^k)/\delta\theta$ is unknown?

Gauss-Newton and the Normal equations

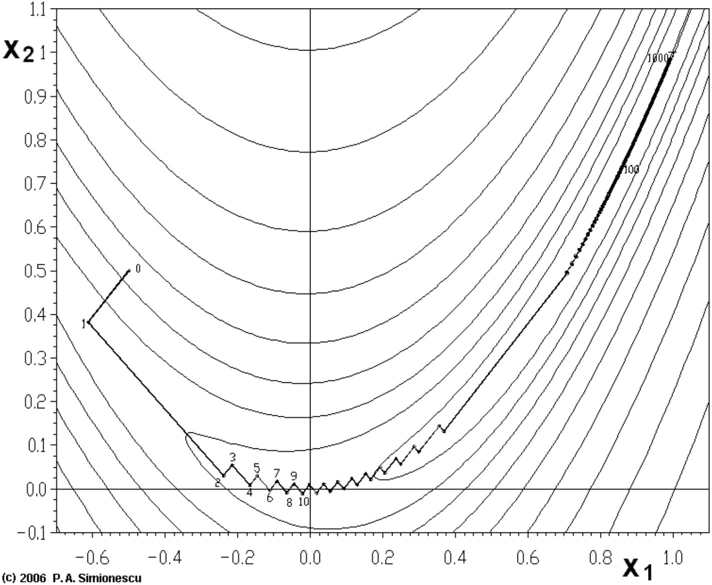
Recall that in multiple *linear* regression, with $y = X\theta + e$ the solution is given by the normal equations

$$X'X\theta = X'y$$

Note that here, the Jacobian of $y - X\theta$ is $-X$, so if we take (arbitrarily) $\theta_0 = (0, 0, ..., 0)'$, then

$$J_f(\theta_k)'J_f(\theta_k)\delta^k = -J_f(\theta_k)f(\theta^k)$$

yields after one step the final solution $\delta^1 = \theta$, as
$(-X)'(-X)\delta = X'y$.
Other starting points yield the same solution for θ.
Further steps will not improve it (i.e., yield $\delta^k = 0$).



(c) 2006 P. A. Simionescu

Problems with steepest descent

- (see previous slide:) steepest descent may be very slow
- Main problem: a minimum may be *local*, other initial values may result in other, better minima
- Cure: apply Gauss-Newton from many different starting points (cumbersome, costly, cpu intensive)
- Global search:
  - apply a grid search – curse of dimensionality. E.g. for three parameters, 50 grid nodes along each direction: $50^3 = 125000$
  - apply random sampling (same problem)
  - use search methods that not *only* go downhill:
    - Metropolis-Hastings (sampling)
    - Simulated Annealing (optimizing)

Metropolis-Hastings

Why would one want probabalistic search?

- global–unlikely areas are searched too (with small probability)
- a probability distribution is richer than a point estimate: Gauss-Newton provides an estimate of $\hat{\theta}$ of θ, given data y. What about the estimation error $\hat{\theta} - \theta$? Second-order derivatives give approximations to standard errors, but not the full distribution.

We explain the simplified version, the Metropolis algorithm

Metropolis algorithm

Given a point in parameter space θ, say $x_t = (\theta_{1,t}, ..., \theta_{p,t})$ we evaluate whether another point, x' is a reasonable alternative. If accepted, we set $x_{t+1} \leftarrow x'$; if not we keep x_t and set $x_{t+1} \leftarrow x_t$.

- if $P(x') > P(x_t)$, we accept x' and set $x_{t+1} = x'$
- if $P(x') < P(x_t)$, then
    - we draw U, a random uniform value from $[0, 1]$, and
    - accept x' if $U < \frac{P(x')}{P(x_t)}$

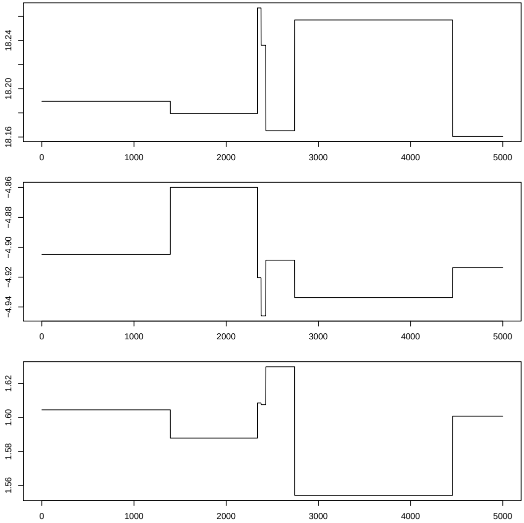
Often, x' is drawn from some normal distribution centered around x_t: $N(x_t, \sigma^2 I)$. Suppose we accept it always, then

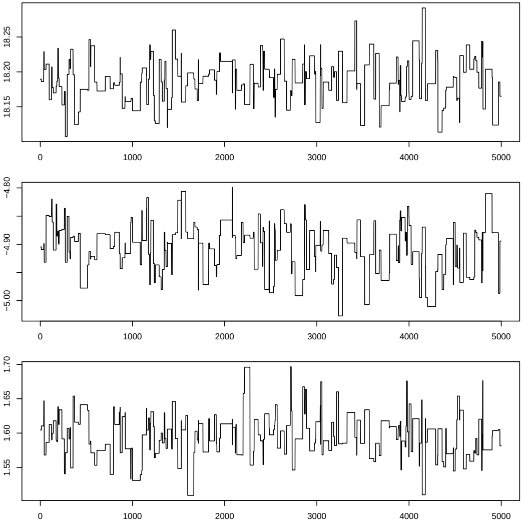
$$x_{t+1} = x_t + e_t$$

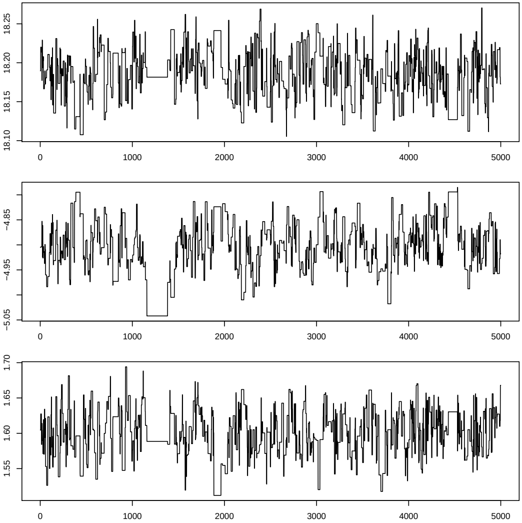
with $e_t \sim N(0, \sigma^2 I)$. Looks familiar?

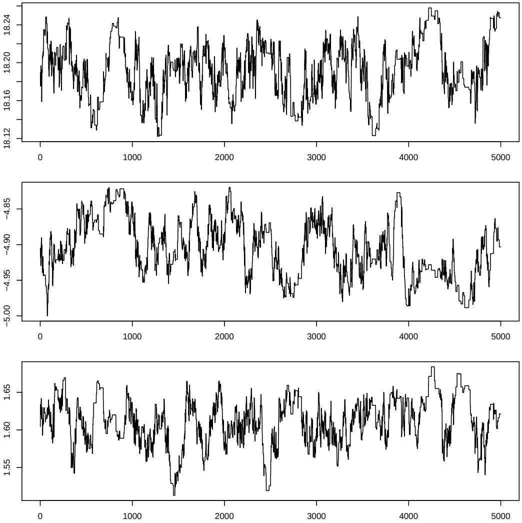
Burn-in, tuning σ^2

- When run for a long time, the Metropolis (and its generalization Metropolis-Hastings) algorithm provide a *correlated* sample of the parameter distribution
- M and MH algorithms provide Markov Chain Monte Carlo samples; another even more popular algorithm is the Gibb's sampler (WinBUGS).
- As the starting value may be quite unlikely, the first part of the chain (burn-in) is usually discarded.
- if σ^2 is too small, the chain mixes too slowly (consecutive samples are too similar, and do not describe the full PDF)
- if σ^2 is too large, most proposal values are not accepted
- often, during burn-in, σ^2 is tuned such that acceptance rate is close to 60%.
- many chains can be run, using different starting values, in parallel


18.16
18.20
18.24
0
1000
2000
3000
4000
5000
−4.94
−4.92
−4.90
−4.88
−4.86
0
1000
2000
3000
4000
5000
1.56
1.58
1.60
1.62
0
1000
2000
3000
4000
5000


18.15
18.20
18.25
0
1000
2000
3000
4000
5000
−5.00
−4.90
−4.80
0
1000
2000
3000
4000
5000
1.55
1.60
1.65
1.70
0
1000
2000
3000
4000
5000


18.25
18.20
18.15
18.10
−4.85
−4.95
−5.05
1.70
1.65
1.60
1.55
0
1000
2000
3000
4000
5000


18.24
18.20
18.16
18.12
0
1000
2000
3000
4000
5000
−4.85
−4.90
−4.95
−5.00
0
1000
2000
3000
4000
5000
1.65
1.60
1.55
0
1000
2000
3000
4000
5000

Likelihood ratio – side track

For evaluating acceptance, the ratio $\frac{P(x')}{P(x_t)}$ is needed, not the individual values.
This means that $P(x')$ and $P(x_t)$ are only needed *up to a normalizing constant*: if we have values $aP(x')$ and $aP(x_t)$, than that is sufficient as a cancels out.
This result is *key* to the reason that MCMC and M-H are the work horse in Bayesian statistics, where $P(x')$ is extremely hard to find because it calls for the evaluation of a very high-dimensional integral (the normalizing constant that makes sure that $P(\cdot)$ is a probability) but $aP(x')$, the likelihood of x given data, is much easier to find!

Likelihood function for normal distribution

Normal probability density function:

$$Pr(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(x-\mu)^2}{2\sigma^2})$$

Likelihood, Multivariate; independent observations:

$$Pr(x_1, x_2, ..., x_p; \mu, \sigma) = \prod_{i=1}^{p} Pr(x_i)$$

which is proportional to

$$\exp(-\frac{\sum_{i=1}^{n}(x_i-\mu)^2}{2\sigma^2})$$

Simulated annealing

Simulated Annealing is a related global search algorithm, does not sample the full parameter distribution but searches for the (global) optimimum.

The analogy with *annealing*, the forming of crystals in a slowly cooling substance, is the following:

The current solution is replaced by a worse "nearby" solution with a certain probability that depends on the the degree to which the "nearby" solution is worse, and on the temperature of the cooling process; this temperature slowly decreases, allowing less and smaller changes.

At the start, temperature is large and search is close to random; when temperature decreases search is more and more local and downhill. Random, uphill jumps prevent SA to fall into a local minimum.

Spatial modelling and spatial interpolation

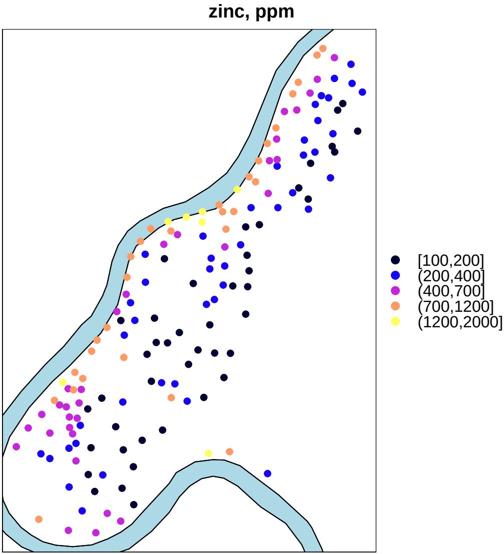
- Simple ways of interpolation
- Simple statistical models for interpolation
- Geostatistical interpolation
- Deterministic models
- Combined approaches

Taking a step back

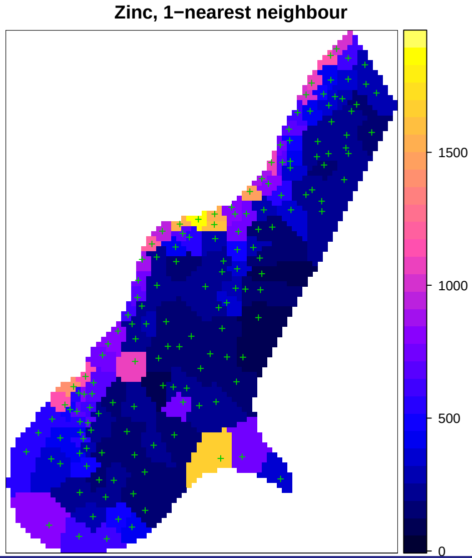
Why do we need models?

- to understand *relations* or *processes*
- to assess (predict, forcast, map) something we do or did not measure *and cannot see*
- to assess the consequence of decisions (scenarios) where we cannot measure

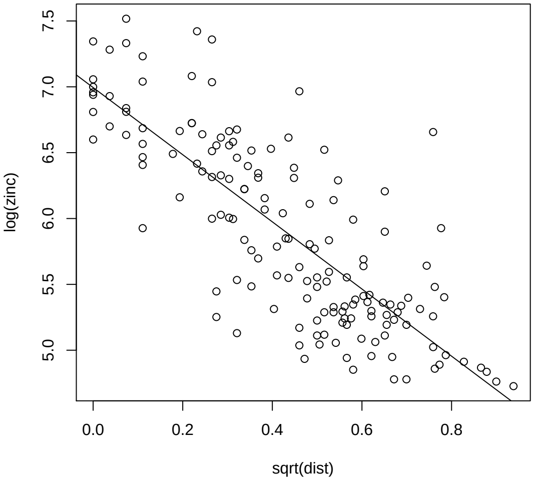
A sample data set



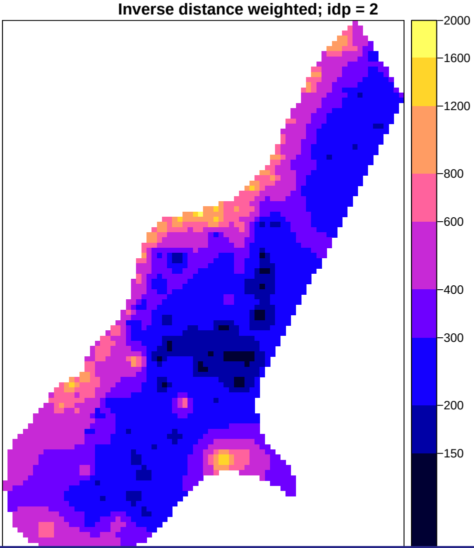
Thiessen "polygons", 1-NN



Zinc concentration vs. distance to river



Simple ways of interpolation



Inverse distance weighted interpolation

Uses a weighted average:

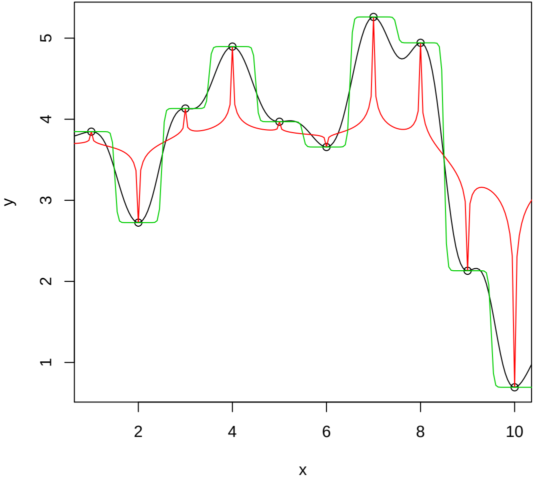
$$\hat{Z}(s_0) = \sum_{i=1}^{n} \lambda_i Z(s_i)$$

with $s_0 = \{x_0, y_0\}$, or $s_0 = \{x_0, y_0, \text{depth}_0\}$ weights inverse proportional to power p of distance:

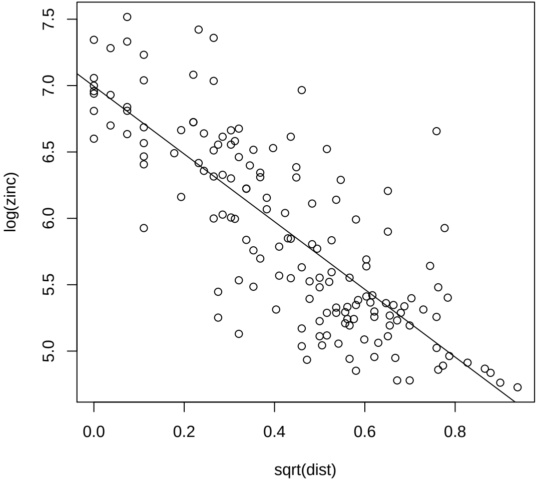
$$\lambda_i = \frac{|s_i - s_0|^{-p}}{\sum_{i=1}^{n} |s_i - s_0|^{-p}}$$

- power p: tuning parameter
- if for some i, $|s_i - s_0| = 0$, then $\lambda_i = 1$ and other weights become zero
- $\Rightarrow$ exact interpolator

Effect of power p



Simple statistical models for interpolation



Time series versus spatial data

Differences:

- spatial data live in 2 (or 3) dimensions
- there's no past and future
- there's no simple conditional independence (AR)

Correspondences

- nearby observations are more alike (auto-correlation)
- we can form moving averages
- coordinate reference systems are a bit like time zones and DST

What information do we have?

- We have measurements $Z(x)$, with x two-dimensional (location on the map)
- we have x and y
- we may have land use data
- we may have soil type or geological data
- we may have remote sensing imagery
- we may have all kinds of relevant information, related to processes that cause (or result from) $Z(x)$
- we have google maps

We don't want to ignore anything important

What information do we have?

- We have measurements $Z(x)$, with x two-dimensional (location on the map)
- we have x and y
- we may have land use data
- we may have soil type or geological data
- we may have remote sensing imagery
- we may have all kinds of relevant information, related to processes that cause (or result from) $Z(x)$
- we have google maps

We don't want to ignore anything important

What information do we have?

- We have measurements $Z(x)$, with x two-dimensional (location on the map)
- we have x and y
- we may have land use data
- we may have soil type or geological data
- we may have remote sensing imagery
- we may have all kinds of relevant information, related to processes that cause (or result from) $Z(x)$
- we have google maps

We don't want to ignore anything important

What information do we have?

- We have measurements $Z(x)$, with x two-dimensional (location on the map)
- we have x and y
- we may have land use data
- we may have soil type or geological data
- we may have remote sensing imagery
- we may have all kinds of relevant information, related to processes that cause (or result from) $Z(x)$
- we have google maps

We don't want to ignore anything important

What information do we have?

- We have measurements $Z(x)$, with x two-dimensional (location on the map)
- we have x and y
- we may have land use data
- we may have soil type or geological data
- we may have remote sensing imagery
- we may have all kinds of relevant information, related to processes that cause (or result from) $Z(x)$
- we have google maps

We don't want to ignore anything important

What information do we have?

- We have measurements $Z(x)$, with x two-dimensional (location on the map)
- we have x and y
- we may have land use data
- we may have soil type or geological data
- we may have remote sensing imagery
- we may have all kinds of relevant information, related to processes that cause (or result from) $Z(x)$
- we have google maps

We don't want to ignore anything important

What information do we have?

- We have measurements $Z(x)$, with x two-dimensional (location on the map)
- we have x and y
- we may have land use data
- we may have soil type or geological data
- we may have remote sensing imagery
- we may have all kinds of relevant information, related to processes that cause (or result from) $Z(x)$
- we have google maps

We don't want to ignore anything important

Regression or correlation?

The power of regression models for spatial prediction

... is hard to overestimate. Regression and correlation are the fork and knife of statistics.

- linear models have endless application: polynomials, interactions, nested effects, ANOVA/ANCOVA models, hypothesis testing, lack of fit testing, ...
- predictors can be transformed non-linearly
- linear models can be generalized: logistic regression, Poisson regression, ..., to cope with discrete data (0/1 data, counts, log-normal)
- many derived techniques solve one particular issue in regression, e.g.:
  - ridge regression solves collinearity (extreme correlation among predictors)
  - stepwise regression automatically selects "best" models among many candidates
  - classification and regression trees

Why is regression difficult in spatial problems?

Regression models assume independent observations. Spatial data are always to some degree spatially correlated. This does not mean we should discard regression, but rather think about

- to which extent is an outcome dependent on independence?
- to which extent is regression *robust* agains a violated assumption of independent observations?
- to which extent *is* the assumption violated? (how strong is the correlation)

Why is regression difficult in spatial problems?

Regression models assume independent observations. Spatial data are always to some degree spatially correlated. This does not mean we should discard regression, but rather think about

- to which extent is an outcome dependent on independence?
- to which extent is regression *robust* agains a violated assumption of independent observations?
- to which extent *is* the assumption violated? (how strong is the correlation)

What is spatial correlation?

Waldo Tobler’s first law in geography:
Everything is related to everything else, but near things are more related than distant things.” [Tobler, 1970, p.236]
TOBLER, W. R. (1970). ”A computer model simulation of urban growth in the Detroit region”. Economic Geography, 46(2): 234-240.
But how then is ”being related” expressed?

What is spatial correlation?

Waldo Tobler's first law in geography:
Everything is related to everything else, but near things are more related than distant things." [Tobler, 1970, p.236]
TOBLER, W. R. (1970). "A computer model simulation of urban growth in the Detroit region". Economic Geography, 46(2): 234-240.
But how then is "being related" expressed?

What is spatial correlation?

Idea from time series: look at lagged correlations, and the h-scatterplot.
What is it? Plots of (or correlation between) $Z(s)$ and $Z(s + h)$, where $s + h$ is s, shifted by h (time distance, spatial distance).

Random variables: expectation, variance, covariance

Random variable: Z follows a probability distribution, which specifies $f(z) = \Pr(Z = z)$ or $F(z) = \Pr(Z \le z)$

Expectation: $\mathrm{E}(Z) = \int_{-\infty}^{\infty} f(s)ds$ – center of mass, mean.

Variance: $\mathrm{Var}(Z) = \mathrm{E}(Z - \mathrm{E}(Z))^2$ – mean squared distance from mean; measure of spread; square root: standard deviation of Z.

Covariance: $\mathrm{Cov}(X, Y) = \mathrm{E}((X - \mathrm{E}(X))(Y - \mathrm{E}(Y)))$ – mean product; can be negative; $\mathrm{Cov}(X, X) = \mathrm{Var}(X)$.

Correlation: $r_{XY} = \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var}(X)\mathrm{Var}(Y)}}$ – normalized $[-1, 1]$ covariance. -1 or $+1$: perfect correlation.

Random variables: expectation, variance, covariance

Random variable: Z follows a probability distribution, which specifies $f(z) = \Pr(Z = z)$ or $F(z) = \Pr(Z \leq z)$

Expectation: $\mathrm{E}(Z) = \int_{-\infty}^{\infty} f(s)ds$ – center of mass, mean.

Variance: $\mathrm{Var}(Z) = \mathrm{E}(Z - \mathrm{E}(Z))^2$ – mean squared distance from mean; measure of spread; square root: standard deviation of Z.

Covariance: $\mathrm{Cov}(X, Y) = \mathrm{E}((X - \mathrm{E}(X))(Y - \mathrm{E}(Y)))$ – mean product; can be negative; $\mathrm{Cov}(X, X) = \mathrm{Var}(X)$.

Correlation: $r_{XY} = \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var}(X)\mathrm{Var}(Y)}}$ – normalized $[-1, 1]$ covariance.

-1 or $+1$: perfect correlation.

Random variables: expectation, variance, covariance

Random variable: Z follows a probability distribution, which specifies $f(z) = \mathrm{Pr}(Z = z)$ or $F(z) = \mathrm{Pr}(Z \leq z)$
Expectation: $\mathrm{E}(Z) = \int_{-\infty}^{\infty} f(s)ds$ – center of mass, mean.
Variance: $\mathrm{Var}(Z) = \mathrm{E}(Z - \mathrm{E}(Z))^2$ – mean squared distance from mean; measure of spread; square root: standard deviation of Z.
Covariance: $\mathrm{Cov}(X, Y) = \mathrm{E}((X - \mathrm{E}(X))(Y - \mathrm{E}(Y)))$ – mean product; can be negative; $\mathrm{Cov}(X, X) = \mathrm{Var}(X)$.
Correlation: $r_{XY} = \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var}(X)\mathrm{Var}(Y)}}$ – normalized $[-1, 1]$ covariance. -1 or $+1$: perfect correlation.

Random variables: expectation, variance, covariance

Random variable: Z follows a probability distribution, which specifies $f(z) = \Pr(Z = z)$ or $F(z) = \Pr(Z \le z)$

Expectation: $\mathrm{E}(Z) = \int_{-\infty}^{\infty} f(s)ds$ – center of mass, mean.

Variance: $\mathrm{Var}(Z) = \mathrm{E}(Z - \mathrm{E}(Z))^2$ – mean squared distance from mean; measure of spread; square root: standard deviation of Z.

Covariance: $\mathrm{Cov}(X, Y) = \mathrm{E}((X - \mathrm{E}(X))(Y - \mathrm{E}(Y)))$ – mean product; can be negative; $\mathrm{Cov}(X, X) = \mathrm{Var}(X)$.

Correlation: $r_{XY} = \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var}(X)\mathrm{Var}(Y)}}$ – normalized $[-1, 1]$ covariance.

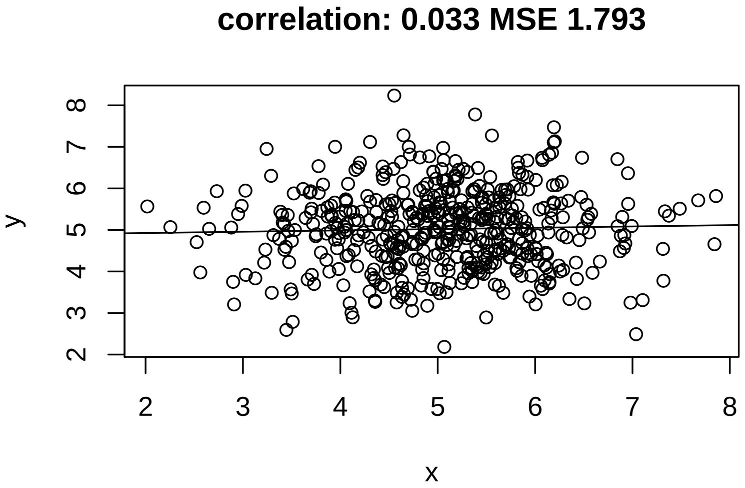
-1 or $+1$: perfect correlation.

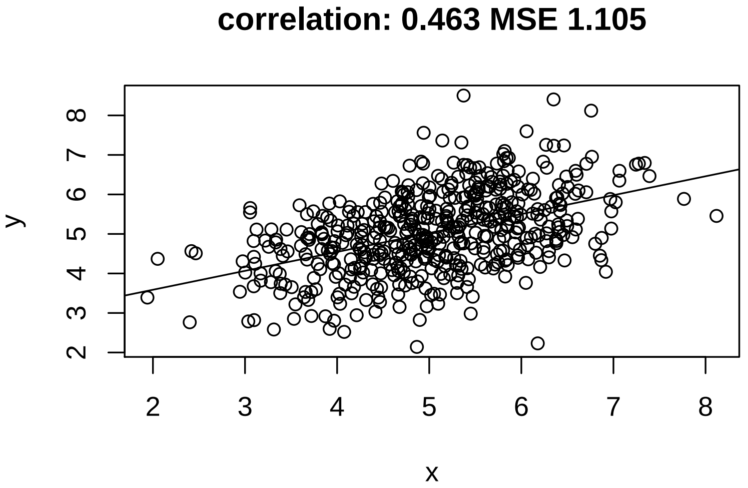
Random variables: expectation, variance, covariance

Random variable: Z follows a probability distribution, which specifies $f(z) = \Pr(Z = z)$ or $F(z) = \Pr(Z \le z)$
Expectation: $\mathrm{E}(Z) = \int_{-\infty}^{\infty} f(s)ds$ – center of mass, mean.
Variance: $\mathrm{Var}(Z) = \mathrm{E}(Z - \mathrm{E}(Z))^2$ – mean squared distance from mean; measure of spread; square root: standard deviation of Z.
Covariance: $\mathrm{Cov}(X, Y) = \mathrm{E}((X - \mathrm{E}(X))(Y - \mathrm{E}(Y)))$ – mean product; can be negative; $\mathrm{Cov}(X, X) = \mathrm{Var}(X)$.
Correlation: $r_{XY} = \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var}(X)\mathrm{Var}(Y)}}$ – normalized $[-1, 1]$ covariance.
-1 or $+1$: perfect correlation.

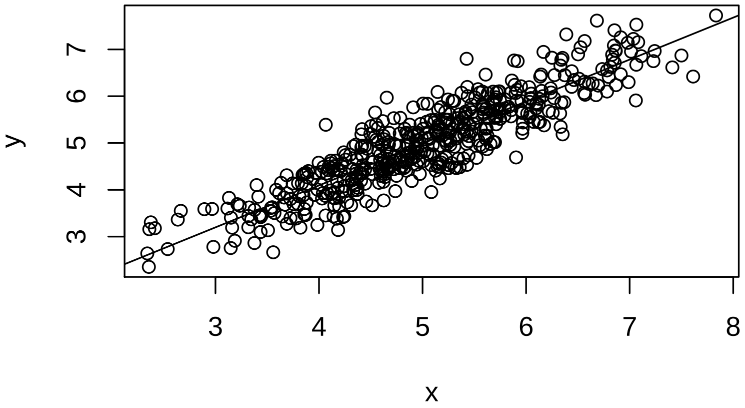
Normal distribution

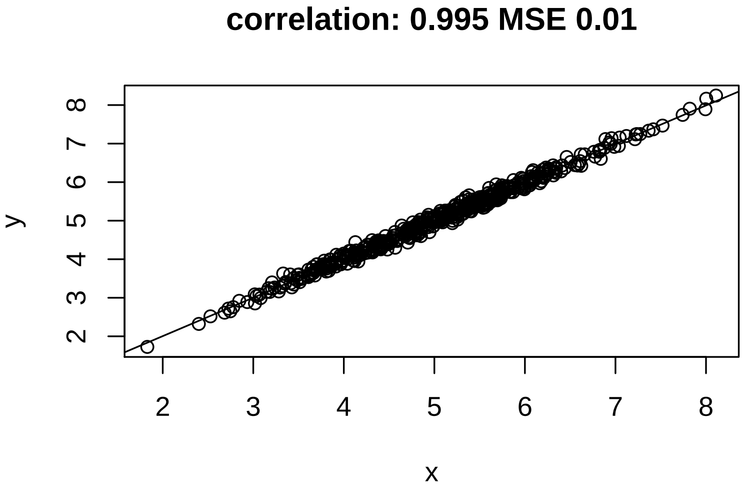
- *univariate*: If Z follows a normal distribution, its probability distribution is *completely* characterized by its mean $E(Z) = \mu$ and variance $\mathrm{Var}(Z) = \sigma^2$
- *multivariate*: If the vector $Z = (Z_1, Z_2, ..., Z_p)$ follows a *multivariate* normal distribution, its marginal distributions are univariate normal, and its *joint* probability distribution is *completely* characterized by the mean vector $E(Z) = \mu = (\mu_1, ...\mu_p)$ and covariance matrix V, of which element (i, j) equals $\mathrm{Cov}(Z_i, Z_j)$
- covariance matrices have variances on the diagonal


correlation: 0.033 MSE 1.793
y
2
3
4
5
6
7
8
x
2
3
4
5
6
7
8


correlation: 0.463 MSE 1.105
y
2
3
4
5
6
7
8
x
2
3
4
5
6
7
8

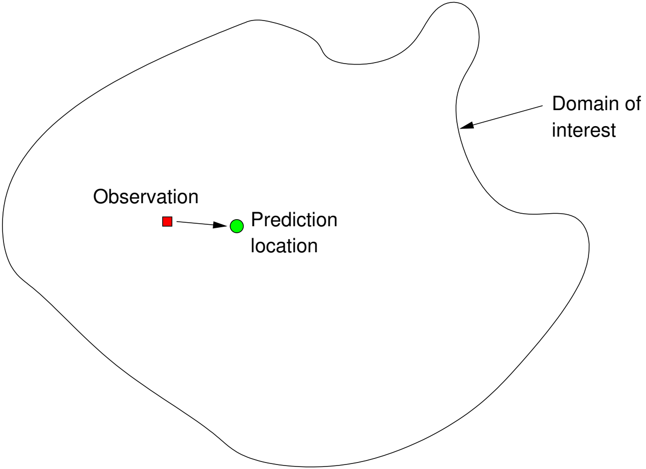
correlation: 0.899 MSE 0.203




correlation: 0.995 MSE 0.01
y
2
3
4
5
6
7
8
x
2
3
4
5
6
7
8

How can correlation help prediction?

Problem:



Questions

Given observation $z(s_1)$, how to predict $z(s_0)$?

- What is the best predicted value at s_0, $\hat{z}(s_0)$?
- How can we compute a measure of error for $\hat{z}(s_0) - z(s_0)$?
- Can we compute e.g. 95% prediction intervals for the unknown $z(s_0)$?

Obviously, given *only* $z(s_1)$, the best predictor for $z(s_0)$ is $\hat{z}(s_0) = z(s_1)$. But what is the error variance, i.e. $\mathrm{Var}(\hat{z}(s_0) - z(s_0))$

Questions

Given observation $z(s_1)$, how to predict $z(s_0)$?

- What is the best predicted value at s_0, $\hat{z}(s_0)$?
- How can we compute a measure of error for $\hat{z}(s_0) - z(s_0)$?
- Can we compute e.g. 95% prediction intervals for the unknown $z(s_0)$?

Obviously, given *only* $z(s_1)$, the best predictor for $z(s_0)$ is $\hat{z}(s_0) = z(s_1)$. But what is the error variance, i.e. $\mathrm{Var}(\hat{z}(s_0) - z(s_0))$

Questions

Given observation $z(s_1)$, how to predict $z(s_0)$?

- What is the best predicted value at s_0, $\hat{z}(s_0)$?
- How can we compute a measure of error for $\hat{z}(s_0) - z(s_0)$?
- Can we compute e.g. 95% prediction intervals for the unknown $z(s_0)$?

Obviously, given *only* $z(s_1)$, the best predictor for $z(s_0)$ is $\hat{z}(s_0) = z(s_1)$. But what is the error variance, i.e. Var$(\hat{z}(s_0) - z(s_0))$

Estimation error

Let both $z(s_1)$ and $z(s_0)$ come from a field that has variance 1, i.e. $\mathsf{Var}(z(s_0)) = \mathsf{Var}(z(s_1)) = 1$, and that has a constant mean: $\mathsf{E}(z(s_0)) = \mathsf{E}(z(s_1)) = m$
Then,

$$\mathsf{Var}(\hat{z}(s_0) - z(s_0)) = \mathsf{Var}(z(s_1) - z(s_0))$$

As both have the same mean, this can be written as

$$\mathsf{E}(\hat{z}(s_0) - z(s_0))^2 = \mathsf{Var}(z(s_1)) + \mathsf{Var}(z(s_0)) - 2\mathsf{Cov}(z(s_1), z(s_0))$$

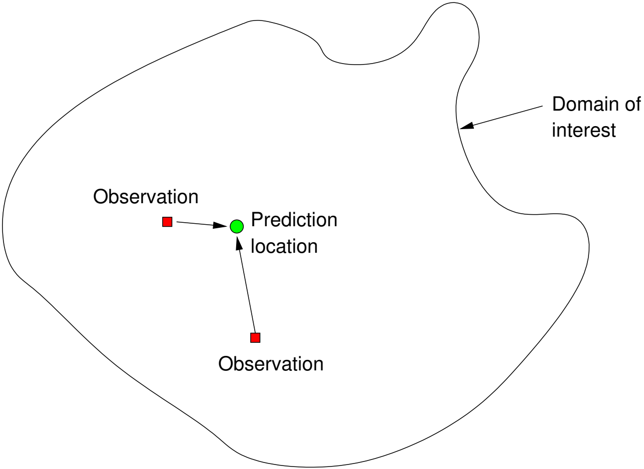
As both have variance 1, this equals $2(1 - r)$ with r the correlation between $z(s_0)$ and $z(s_1)$. Examples follow.

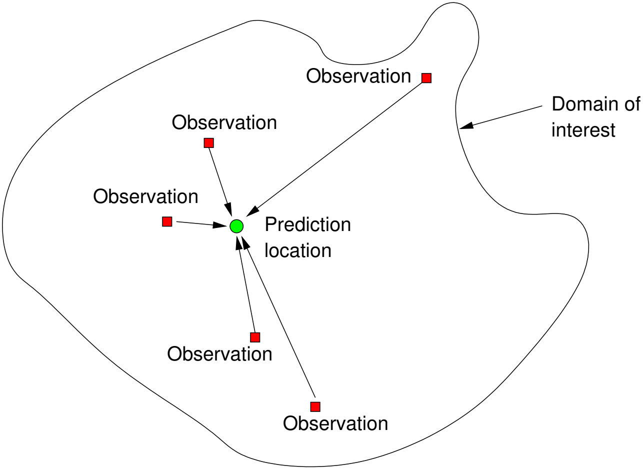
Suppose we know the mean

If we know the mean μ, it may be a good idea to use a compromise between the observation and the mean, e.g.

$$\hat{z}(s_0) = (1 - r)\mu + rz(s_1)$$

Next problems...




Observation
Observation
Domain of interest
Observation
Prediction location
Observation
Observation

What is Geostatistical Interpolation?

Geostatistical interpolation (kriging) uses linear predictors

$$\hat{z}(s_0) = \sum_{i=1}^{n} \lambda_i z(s_i)$$

with weights chosen such that

- the interpolated values is unbiased: $\mathsf{E}(\hat{z}(s_0) - z(s_0)) = 0$ and
- has mininum variance: $\mathsf{Var}(\hat{z}(s_0) - z(s_0))$ is at minimum.

All that is needed is variances and correlations.

Random variables

Random variables (RVs) are numeric variables whose outcomes are subject to chance.

The cumulative distribution of probability $F_x(\cdot)$ over outcomes z over all possible values of the RV Z is the probability distribution function:

$$P(Z \leq z) = F_Z(z) = \int_{-\infty}^{z} f_Z(u)du$$

where $f_Z(\cdot)$ is the probability *density* function of Z. The sum of all probability is 1.

Random variables have an expectation (mean):

$E(Z) = \int_{-\infty}^{\infty} u f_Z(u)du$ and a variance:

$\mathrm{Var}(Z) = E[(Z - E(Z))^2]$.

Try to think of $E(Z)$ as $\frac{1}{n}\sum_{i=1}^{n} z_i$, with $i \rightarrow \infty$.

Two random variables X and Y have covariance defined as

$\mathrm{Cov}(X, Y) = E[(X - E(X))(Y - E(Y))]$

Correlation and covariance

Correlation is scaled covariance, scaled by the variances. For two variables X and Y, it is

$$\mathrm{Corr}(X,Y) = \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var}(X)\mathrm{Var}(Y)}}$$

It is quite easy to show that $|\mathrm{Cov}(X,Y)| \le \sqrt{\mathrm{Var}(X)\mathrm{Var}(Y)}$, so correlation ranges from -1 to 1. For this, note that $\mathrm{Cov}(X,X) = \mathrm{Var}(X)$. and $\mathrm{Cov}(X,-X) = -\mathrm{Var}(X)$.

It is perhaps easier to think of covariance as unscaled correlation.

A large covariance does not imply a strong correlation

Correlation and covariance

Correlation is scaled covariance, scaled by the variances. For two variables X and Y, it is

$$\mathrm{Corr}(X,Y) = \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var}(X)\mathrm{Var}(Y)}}$$

It is quite easy to show that $|\mathrm{Cov}(X,Y)| \leq \sqrt{\mathrm{Var}(X)\mathrm{Var}(Y)}$, so correlation ranges from -1 to 1. For this, note that $\mathrm{Cov}(X,X) = \mathrm{Var}(X)$. and $\mathrm{Cov}(X,-X) = -\mathrm{Var}(X)$.

It is perhaps easier to think of covariance as unscaled correlation.

A large covariance does not imply a strong correlation

Correlation and covariance

Correlation is scaled covariance, scaled by the variances. For two variables X and Y, it is

$$\mathrm{Corr}(X,Y) = \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var}(X)\mathrm{Var}(Y)}}$$

It is quite easy to show that $|\mathrm{Cov}(X,Y)| \le \sqrt{\mathrm{Var}(X)\mathrm{Var}(Y)}$, so correlation ranges from -1 to 1. For this, note that $\mathrm{Cov}(X,X) = \mathrm{Var}(X)$. and $\mathrm{Cov}(X,-X) = -\mathrm{Var}(X)$.
It is perhaps easier to think of covariance as unscaled correlation.
A large covariance does not imply a strong correlation

The quadratic form

We will not consider single random variables (how boring), but rather large collections of them. In fact, we will consider each observation $z(s_i)$ as a realisation (outcome) of a random variable $Z(s_i)$, and consider the Z variable at all other locations also as separate random variables, say $Z(s_0)$ for any s_0 in the domain of interest.

Let $Z = [Z(s_1)\ Z(s_2)\ ...\ Z(s_n)]'$ then $\mathrm{Var}(Z)$ is the covariance matrix of vector Z, with i,j-th element $\mathrm{Cov}(Z(s_i), Z(s_j))$, implying it has variances on the diagonal.

Then, it is easy to show that for non-random weights $\lambda = [\lambda_1 ... \lambda_n]'$ the quadratic form $\lambda' Z = \sum_{i=1}^{n} \lambda_i Z(s_i)$ has variance

$$\mathrm{Var}(\lambda' Z) = \lambda' \mathrm{Var}(Z) \lambda = \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j \mathrm{Cov}(Z(s_i), Z(s_j)) = \lambda' V \lambda$$

Why do we need this?

When we predict (interpolate), we're forming linear combinations, $\sum_{i=1}^n \lambda_i Z(s_i)$, and want to know the variance of $\sum_{i=1}^n \lambda_i Z(s_i) - Z(s_0)$, the interpolation error variance. Only then can we find weights such that it is minimum.
What is the scalar $\mathrm{Var}(\sum_{i=1}^n \lambda_i Z(s_i) - Z(s_0))$? Write as

$$\mathrm{Var}(\lambda' Z - Z(s_0)) = \mathrm{Var}(\lambda' Z) + \mathrm{Var}(Z(s_0)) - 2\mathrm{Cov}(\lambda' Z, Z(s_0))$$

$$= \lambda' V \lambda + \sigma_0^2 + \sum_{i=1}^n \lambda_i \mathrm{Cov}(Z(s_i), Z(s_0))$$

with $\sigma_0^2 = \mathrm{Var}(Z(s_0))$
SO,we need variances of all $Z(s_i)$, including for all s_0, and all covariances between pairs $Z(s_i)$ and $Z(s_j)$, including all s_0.

Suppose we know all that

Kriging: find weights λ such that $\mathrm{Var}(Z(s_0) - \hat{Z}(s_0)) = \mathrm{Var}(Z(s_0) - \sum_{i=1}^{n} \lambda_i Z(s_i))$ is minimized, and we have the best (minimum variance) linear predictor.

Best linear prediction weights: Let $V = \mathrm{Var}(Z) \quad (n \times n)$ and $v = \mathrm{Cov}(Z(s_0), Z) \quad (n \times 1)$, and scalar $\mathrm{Var}(Z(s_0)) = \sigma_0^2$.

Expected squared prediction error $\mathrm{E}(Z(s_0) - \hat{Z}(s_0))^2 = \sigma^2(s_0)$

Replace Z with $Z - \mu$ (or assume $\mu = 0$)

$\sigma^2(s_0) = \mathrm{E}(Z(s_0) - \lambda' Z)^2 =$

$\mathrm{E}(Z(s_0))^2 - 2\lambda' \mathrm{E}(Z(s_0)Z) + \lambda' \mathrm{E}(ZZ')\lambda$

$= \mathrm{Var}(Z(s_0)) - 2\lambda' \mathrm{Cov}(Z(s_0), Z) + \lambda' \mathrm{Var}(Z)\lambda = \sigma_0^2 - 2\lambda' v + \lambda' V \lambda$

Choose λ such that $\frac{\delta \sigma^2(s_0)}{\delta \lambda} = -2v' + 2\lambda' V = 0$

$\lambda' = v' V^{-1}$

BLP/Simple kriging:

$\hat{Z}(s_0) = \mu + v' V^{-1}(Z - \mu) \qquad \sigma^2(s_0) = \sigma_0^2 - v' V^{-1} v$

Unknown, constant mean

Suppose the mean is constant, but not known. This is the most simple *realistic* scenario. We can estimate it from the data, taking into account their covariance (i.e., using weighted averaging):

$$\hat{m} = (\mathbf{1}'V^{-1}\mathbf{1})^{-1}\mathbf{1}'V^{-1}Z$$

with $\mathbf{1}$ a conforming vector with ones, and substitute this mean in the SK prediction equations: BLUP/Ordinary kriging:

$$\hat{Z}(s_0) = \hat{m} + v'V^{-1}(Z - \hat{m})$$

$$\sigma^2(s_0) = \sigma_0^2 - v'V^{-1}v + Q$$

with $Q = (1 - \mathbf{1}'V^{-1}v)'(\mathbf{1}'V^{-1}\mathbf{1})^{-1}(1 - \mathbf{1}'V^{-1}v)$

Stationarity 1

Given prediction location s_0, and data locations s_1 and s_2, we need: $\mathrm{Var}(Z(s_0))$, $\mathrm{Var}(Z(s_1))$, $\mathrm{Var}(Z(s_2))$, $\mathrm{Cov}(Z(s_0), Z(s_1))$, $\mathrm{Cov}(Z(s_0), Z(s_2))$, $\mathrm{Cov}(Z(s_1), Z(s_2))$.
How to get these covariances?

- given a single measurement $z(s_1)$, we can not infer $\mathrm{Var}(Z(s_1))$
- given two measurements $z(s_1)$ and $z(s_2)$, we can *never* infer $\mathrm{Cov}(Z(s_1), Z(s_2))$
- geven a time series at s_1 and s_2, we can infer $\mathrm{Cov}(Z(s_1), Z(s_2))$, but how to infer $\mathrm{Cov}(Z(s_0), Z(s_1))$ and $\mathrm{Cov}(Z(s_0), Z(s_2))$?

Solution: assume stationarity.

Stationarity 2

Stationarity of the

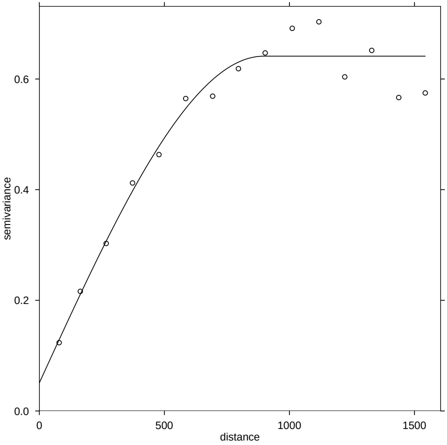
mean	$\mathrm{E}(Z(s_1)) = \mathrm{E}(Z(s_2)) = ... = m$
variance	$\mathrm{Var}(Z(s_1)) = \mathrm{Var}(Z(s_2)) = ... = \sigma_0^2$
covariance	$\mathrm{Cov}(Z(s_1), Z(s_2)) = \mathrm{Cov}(Z(s_3), Z(s_4))$ if $s_1 - s_2 = s_3 - s_4$: distance/direction dependence

Second order stationarity: $\mathrm{Cov}(Z(s), Z(s+h)) = C(h)$
which implies: $\mathrm{Cov}(Z(s), Z(s)) = \mathrm{Var}(Z(s)) = C(0)$
The function $C(h)$ is the covariogram of the random function $Z(s)$

From covariance to semivariance

Covariance:
$\mathrm{Cov}(Z(s), Z(s+h)) = C(h) = \mathrm{E}[(Z(s)-m)(Z(s+h)-m)]$
Semivariance: $\gamma(h) = \frac{1}{2}\mathrm{E}[(Z(s)-Z(s+h))^2]$
$\mathrm{E}[(Z(s)-Z(s+h))^2] = \mathrm{E}[(Z(s))^2 + (Z(s+h))^2 - 2Z(s)Z(s+h)]$
[Assume $m=0$]:
$\mathrm{E}[(Z(s)-Z(s+h))^2] = \mathrm{E}[(Z(s))^2] + \mathrm{E}[(Z(s+h))^2] - 2\mathrm{E}[Z(s)Z(s+h)] = 2\mathrm{Var}(Z(s)) - 2\mathrm{Cov}(Z(s), Z(s+h)) = 2C(0) - 2C(h)$
$\gamma(h) = C(0) - C(h)$
$\gamma(h)$ is the semivariogram of $Z(s)$.

The *Variogram*



The *Variogram*

- the central tool to geostatistics
- like a mean squares (variance) in analysis of variance, like a t to a t-test
- measures spatial correlation
- subject to debate: it involves *modelling*
- synonymous to *semivariogram*, but
- semivariance is *not* synonymous to variance

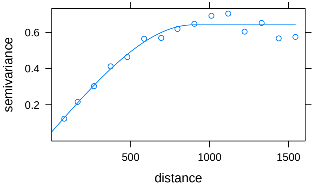
Variogram: how to compute

average squared differences:

$$\hat{\gamma}(\tilde{h}) = \frac{1}{2N_h} \sum_{i=1}^{N_h} (Z(s_i) - Z(s_i + h))^2 \quad h \in \tilde{h}$$

- divide by $2N_h$:
  - if finite, $\gamma(\infty) = \sigma^2$
  - *semi*variance
- if data are not gridded, group N_h pairs $s_i, s_i + h$ for which $h \in \tilde{h}$, $\tilde{h} = [h_1, h_2]$
- choose about 10-25 distance intervals $\tilde{h}$, from length 0 to about on third of the area size
- "plot" $\tilde{h}$ at the average value of all $h \in \tilde{h}$

Variogram: terminology



gstat coding (R):

```
> vgm(psill = 0.6, model = "Sph", range = 900, nugget = 0.06)
  model psill range
1   Nug  0.06     0
2   Sph  0.60   900
> vgm(0.6, "Sph", 900, 0.06)
  model psill range
1   Nug  0.06     0
2   Sph  0.60   900
```

Why prefer the variogram over the covariogram

Covariance:
$\mathrm{Cov}(Z(s), Z(s+h)) = C(h) = \mathrm{E}[(Z(s)-m)(Z(s+h)-m)]$
Semivariance: $\gamma(h) = \frac{1}{2}\mathrm{E}[(Z(s)-Z(s+h))^2]$
$\gamma(h) = C(0) - C(h)$

- tradition
- $C(h)$ needs (an estimate of) m, $\gamma(h)$ does not
- $C(0)$ may not exist (∞!), when $\gamma(h)$ does (e.g., Brownian motion)
- *software* wants $\gamma(h)$.

Known, varying mean

This is nothing else then simple kriging, except that the mean is no longer constant; BLP/Simple kriging:

$$\hat{Z}(s_0) = \mu(s_0) + v'V^{-1}(Z - \mu(s))$$

$$\sigma^2(s_0) = \sigma_0^2 - v'V^{-1}v$$

with $\mu(s) = (\mu(s_1), \mu(s_2), ..., \mu(s_n))'$.

Unknown, varying mean

For this, we need to know how the mean varies. Suppose we model this as a linear regression model in p known predictors:

$$Z(s_i) = \sum_{j=0}^{p} \beta_j X_j(s_i) + e(s_i)$$

$$Z(s) = \sum_{j=0}^{p} \beta_j X_j(s) + e(s) = X(s)\beta + e(s)$$

with $X(s)$ the matrix with predictors, and row i and column j containing $X_j(s_i)$, and with $\beta = (\beta_0, ...\beta_p)$. Usually, the first column of X contains zeroes in which case β_0 is an intercept.

Unknown, varying mean (2)

Predictor:

$$\hat{Z}(s_0) = x(s_0)\hat{\beta} + v'V^{-1}(Z - X\hat{\beta})$$

with $x(s_0) = (X_0(s_0), ..., X_p(s_0))$ and $\hat{\beta} = (X'V^{-1}X)^{-1}X'V^{-1}Z$ it has prediction error variance

$$\sigma^2(s_0) = \sigma_0^2 - v'V^{-1}v + Q$$

with $Q = (x(s_0) - X'V^{-1}v)'(X'V^{-1}X)^{-1}(x(s_0) - X'V^{-1}v)$
This form is called external drift kriging, universal kriging or sometimes regression kriging.
Example in `meuse` data set: `log(zinc)` depending on `sqrt(meuse)`

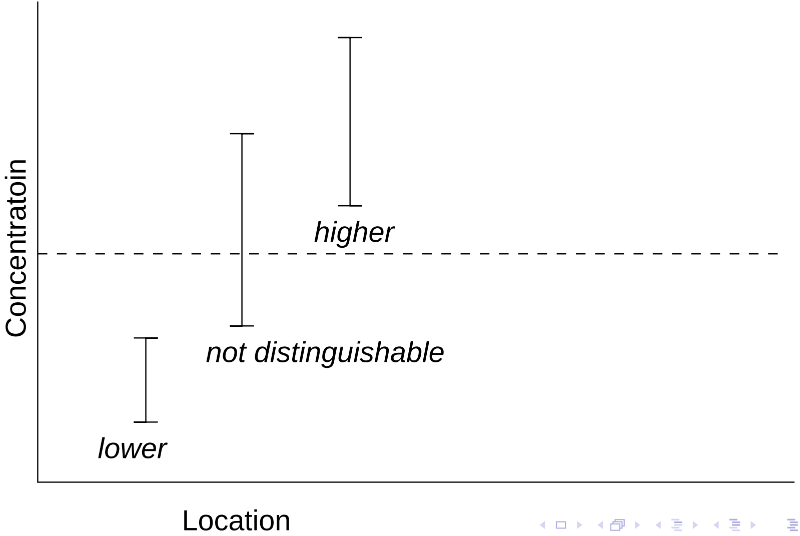
Estimating spatial correlation under the UK model

As opposed to the ordinary kriging model, the universal kriging model needs knowledge of the mean vector in order to estimate the semivariance (or covariance) from the residual vector:

$$\hat{e}(s) = Z(s) - X\hat{\beta}$$

but how to get $\hat{\beta}$ without knowing V? This is a chicken-egg problem. The simplest, but not best, solution is to plug $\hat{\beta}_{OLS}$ in, and from the $e_{OLS}(s)$, estimate V (i.e., the variogram of Z)

Spatial Prediction



Kriging varieties

- Simple kriging: $Z(s) = \mu + e(s)$, μ known
- Ordinary kriging: $Z(s) = m + e(s)$, m unknown
- Universal kriging: $Z(s) = X\beta + e(s)$, β unknown
- SK: linear predictor $\lambda' Z$ with λ such that $\sigma^2(s_0) = \mathrm{E}(Z(s_0) - \lambda' Z)^2$ is minimized
- OK: linear predictor $\lambda' Z$ with λ such that it
  1. has minimum variance $\sigma^2(s_0) = \mathrm{E}(Z(s_0) - \lambda' Z)^2$, and
  2. is unbiased $\mathrm{E}(\lambda' Z) = m$
- second constraint: $\sum_{i=1}^{n} \lambda_i = 1$, weights sum to one.

- UK:
$$\hat{Z}(s_0) = x(s_0)\hat{\beta} + v'V^{-1}(Z - X\hat{\beta})$$
with $x(s_0) = (X_0(s_0), ..., X_p(s_0))$ and
$\hat{\beta} = (X'V^{-1}X)^{-1}X'V^{-1}Z$
$$\sigma^2(s_0) = \sigma_0^2 - v'V^{-1}v + Q$$
with $Q = (x(s_0) - X'V^{-1}v)'(X'V^{-1}X)^{-1}(x(s_0) - X'V^{-1}v)$
- OK: fill in a column vector with ones for X: $X = (1, 1, ..., 1)'$ and $X_0 = 1$
- SK: take out the trend/unknown mean

UK and linear regression

If Z has no spatial correlation, all covariances are zero and $v = 0$ and $V = \text{diag}(\sigma^2)$. This implies that

$$\hat{Z}(s_0) = x(s_0)\hat{\beta} + v'V^{-1}(Z - X\hat{\beta})$$

with $\hat{\beta} = (X'V^{-1}X)^{-1}X'V^{-1}Z$ reduces to

$$\hat{Z}(s_0) = x(s_0)\hat{\beta}$$

with $\hat{\beta} = (X'X)^{-1}X'Z$, i.e., ordinary least squares regression. Note that

- under this model the residual does not carry information, as it is white noise
- in spatial prediction, UK can not be worse than linear regression, as linear regression is a limiting case of a more general model.

Global vs. local predictors

In many cases, instead of using all data, the number of observations used for prediction are limited to a selection of nearest observations, based on

- number of observations or
- distance to prediction location s_0
- possibly, in addition, directions

The reason for this is usually either

- statistical, allowing for a more flexible mean/trend structure
- practical, if n gets large

Statistical arguments for local prediction

- estimating β locally instead of globally means that
  - β will adjust to local situations (less bias)
  - it will be harder to estimate β from less information, so (slightly?) larger prediction errors will result (larger variance)
  - X needs to be non-singular in every neighbourhood
- some authors claim that local trends are so adaptive, that one can ignore spatial correlation of the residual
- Using local linear regression with *weights* that decay with distance is called *geographically weighted regression*, GWR

Practical arguments for local prediction

- The number of observations, n may become very large.
  - lidar data, 3D chemical, satellite sensors, geotechnical, seismic, ...
- Computing $V^{-1}v$ is the expensive part; it is $O(N^2)$ or $O(N^3)$ as V is usually not of simple structure
- there is a trade-off; for a global neighbourhood, the expensive part, factoring V needs only be done once, for a local neighbourhood for each unique neighbourhood (in practice: for each s_0).
- selecting local neighbourhoods also costs time; naive selection $O(n \log n)$ doesn't scale well
- `gstat` uses quadtrees/octtrees, inspired by `http://donar.umiacs.umd.edu/quadtree/index.html`

Predicting block means

Instead of predicting $Z(s_0)$ for a "point" location, one might be interested at predicting the average of $Z(s)$ over a block, B_0, i.e.

$$Z(B_0) = \frac{1}{|B_0|} \int_{B_0} Z(u) du$$

- This can (naively) be done by predicting Z over a large number of points s_0 inside B_0, and averaging
- For the prediction *error*, of $\hat{Z}(B_0)$, we then need the covariances between all point predictions
- a more efficient way is to use block kriging, which does both at once

Reason why one wants block means

Examples

- mining: we cannot mine point values
- soil remediation: we cannot remediate points
- RS: we can match satellite image pixels
- disaster management: we cannot evacuate points
- environment: legislation may be related to blocks
- accuracy: block means can be estimated with smaller errors than points

cokriging

Cokriging sets the multivariate equivalent of kriging, which is, in terms of number of dependent variables, univariate. Kriging:

$$Z(s) = X(s)\beta + e(s)$$

Cokriging:

$$Z_1(s) = X_1(s)\beta_1 + e_1(s)$$

$$Z_2(s) = X_2(s)\beta_2 + e_2(s)$$

$$Z_k(s) = X_k(s)\beta_k + e_k(s)$$

with $V = \mathrm{Cov}(e_1, e_2, ..., e_k)$

Cases where this is useful: multiple spatial correlated variables such as

- chemical properties (auto-analyzers!)
- sediment composition
- electromagnetic spectra (imagery/remote sensing)
- ecological data (abiotic factors; species abundances)
- (space-time data, with discrete time)

Two types of applications:

- undersampled case: secondary variables help prediction of a primary, because we have more samples of them (image?)
- equally sampled case: secondary variables don't help prediction much, but we are interested in *multivariate prediction*, i.e. prediction error covariances.

Cokriging prediction

Cokriging prediction is not substantially different from kriging prediction, it is just a lot of book-keeping.
Multivariable prediction involves the joint prediction of multiple, both spatially and cross-variable correlated variables. Consider m distinct variables, and let $\{Z_i(s), X_i, \beta^i, e_i(s), x_i(s_0), v_i, V_i\}$ correspond to $\{Z(s), X, \beta, e(s), x(s_0), v, V\}$ of the i-th variable.
Next, let $\mathbf{Z}(s) = (Z_1(s)', ..., Z_m(s)')'$, $\mathbf{B} = ({\beta^1}', ..., {\beta^m}')'$, $\mathbf{e}(s) = (e_1(s)', ..., e_m(s)')'$,

$$\mathbf{X} = \left[\begin{array}{cccc} X_1 & 0 & ... & 0 \\ 0 & X_2 & ... & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & ... & X_m \end{array} \right], \ \mathbf{x}(s_0) = \left[\begin{array}{cccc} x_1(s_0) & 0 & ... & 0 \\ 0 & x_2(s_0) & ... & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & ... & x_m(s_0) \end{array} \right]$$

with 0 conforming zero matrices, and

$$\mathbf{v} = \begin{bmatrix} v_{1,1} & v_{1,2} & \dots & v_{1,m} \\ v_{2,1} & v_{2,2} & \dots & v_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ v_{m,1} & v_{m,2} & \dots & v_{m,m} \end{bmatrix}, \ \mathbf{V} = \begin{bmatrix} V_{1,1} & V_{1,2} & \dots & V_{1,m} \\ V_{2,1} & V_{2,2} & \dots & V_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ V_{m,1} & V_{m,2} & \dots & V_{m,m} \end{bmatrix}$$

where element i of $v_{k,l}$ is $\mathrm{Cov}(Z_k(s_i), Z_l(s_0))$, and where element (i,j) of $V_{k,l}$ is $\mathrm{Cov}(Z_k(s_i), Z_l(s_j))$.
The multivariable prediction equations equal the previous UK equations and when all matrices are substituted by their multivariable forms (see also Ver Hoef and Cressie, Math.Geol., 1993), and when for σ_0^2, Σ is substituted with $\mathrm{Cov}(Z_i(s_0), Z_j(s_0))$ in its (i,j)-th element. Note that the prediction variance is now a prediction error covariance matrix.

What is needed?

The main tool for estimating semivariances between different variables is the *cross variogram*, defined for collocated data as

$$\gamma_{ij}(h) = \mathsf{E}[(Z_i(s) - Z_i(s+h))(Z_j(s) - Z_j(s+h))]$$

and for non-collocated data as

$$\gamma_{ij}(h) = \mathsf{E}[(Z_i(s) - m_i)(Z_j(s) - m_j)]$$

with m_i and m_j the means of the respective variables. Sample cross variograms are the obvious sums over the available pairs or cross pairs, as in one of

$$\hat{\gamma}_{jk}(\tilde{h}) = \frac{1}{N_h} \sum_{i=1}^{N_h} (Z_j(s_i) - Z_j(s_i+h))(Z_k(s_i) - Z_k(s_i+h))$$

$$\hat{\gamma}_{jk}(\tilde{h}) = \frac{1}{N_h} \sum_{i=1}^{N_h} (Z_j(s_i) - m_j)(Z_k(s_i+h) - m_k)$$

Permissible cross covariance functions

Two classes of permissible cross covariance (semivariance) functions are often used:

- intrinsic correlation (IC):

  $$\gamma_{jk}(h) = \alpha_{jk}\sqrt{\gamma_{jj}(h)\gamma_{kk}(h)}$$

  parameters α_{jk} are correlation cofficients; very strict
- linear model of coregionalization (LMC):

  $$\gamma_{jk}(h) = \sum_{l=1}^{p} \gamma_{jk,p}(h)$$

  (e.g., nugget + spherical model), and

  $$\gamma_{jk,p}(h) = \alpha_{jk,p}\sqrt{\gamma_{jj,p}(h)\gamma_{kk,p}(h)}$$

How to do this?

As multivariable analysis may involve numerous variables, we need to start organising the available information. For that reason, we collect all the observation data specifications in a gstat object, created by the function gstat. This function does nothing else than ordering (and actually, copying) information needed later in a single object. Consider the following definitions of four heavy metals:

```
g <- gstat(NULL, "logCd", log(cadmium)˜1, meuse)
g <- gstat(g, "logCu", log(copper)˜1, meuse)
g <- gstat(g, "logPb", log(lead)˜1, meuse)
g <- gstat(g, "logZn", log(zinc)˜1, meuse)
g
vm <- variogram(g)
vm.fit <- fit.lmc(vm, g, vgm(1, "Sph", 800, 1))
plot(vm, vm.fit)
```

Kriging predictions and errors – how good are they?

Cross validation can be used to assess the quality of any interpolation, including kriging. We split the data set in n parts (folds). For each part, we

- leave out the observations of this fold
- use the observations of all other folds to predict the values at the locations of this fold
- compare the predictions with the observations

This is called n-fold cross validation. If n equals the number of observation, it is called leave-one-out cross validation (LOOCV).

Cross validation: what does it yield?

- residuals $r(s_i) = z(s_i) - \hat{z}(s_i)$ – histograms, maps, summary statistics
- mean residual should be near zero
- mean square residual $\sum r(s_i)^2$ should be as small as possible

In case the interpolation method yields a prediction error we can compute z-scores: $r(s_i)/\sigma(s_i)$
The z-score allows the validation of the kriging error, as the z-score should have mean close to zero and variance close to 1. If the variance of the z-score is larger (smaller) than 1, the kriging standard error is underestimating (overestimating) the true interpolation error, on average.

Kriging errors – so what?

Suppose legislation prescribes remediation in case zinc exceeds 500 ppm. Where does the zinc level exceed 500 ppm?

- we can compare the map of the predictions with 500. However:
  - $\hat{z}(s_0)$ does not equal $z(s_0)$:
  - $\hat{z}(s_0)$ is more smooth than $z(s_0)$
  - $\hat{z}(s_0)$ is closer to the mean than $z(s_0)$
  - smoothing effect is stronger if spatial correlation is small or nugget effect is relatively large
- alternatively we can assume that the true (unknown) value follows a probability distribution, with mean $\hat{z}(s_0)$ and standard error $\sigma(s_0)$.
- this latter approach acknowledges that $\sigma(s_0)$ is useful as a measure of interpolation accuracy

Kriging errors – so what?

Suppose legislation prescribes remediation in case zinc exceeds 500 ppm. Where does the zinc level exceed 500 ppm?

- we can compare the map of the predictions with 500. However:
  - $\hat{z}(s_0)$ does not equal $z(s_0)$:
  - $\hat{z}(s_0)$ is more smooth than $z(s_0)$
  - $\hat{z}(s_0)$ is closer to the mean than $z(s_0)$
  - smoothing effect is stronger if spatial correlation is small or nugget effect is relatively large
- alternatively we can assume that the true (unknown) value follows a probability distribution, with mean $\hat{z}(s_0)$ and standard error $\sigma(s_0)$.
- this latter approach acknowledges that $\sigma(s_0)$ is useful as a measure of interpolation accuracy

Conditional probability

- we can use e.g. the normal distribution (on the log-scale?) to assess the conditional probability $\Pr(Z(s_0) > 500 | z(s_1), ..., z(s_n))$
- the additional assumption underlying this is *multivariate normality*: in addition to having stationary mean and covariance, the field Z is now assumed to follow a stationary, multivariate normal distribution. This means that any single $Z(s_i)$ follows a normal distribution, and any pair $Z(s_i), Z(s_j)$ follows a bivariate normal distribution, with known variances and covariance.

How?

```
out = krige(log(zinc)~1, meuse, meuse.grid, v.fit)
out$p500 = 1 - pnorm(log(500), out$var1.pred, sqrt(out$var1.var)) #$
spplot(out["p500"], col.regions = bpy.colors())
```

Indicator kriging

After transforming the data into indicators, as

$$I(Z(s), 500) = \begin{cases} 1 & \text{if } Z(s) \leq 500, \\ 0 & \text{otherwise} \end{cases}$$

one may krige the indicator (0/1) values, and try to interpret the outcomes as estimated probabilities:

$$\hat{I}(Z(s_0), 500) = \hat{\Pr}(Z(s_0) < 500)$$

Indicator kriging

Advantage:

- easy!
- non-parametric: we do not need to assume a Gaussian distribution

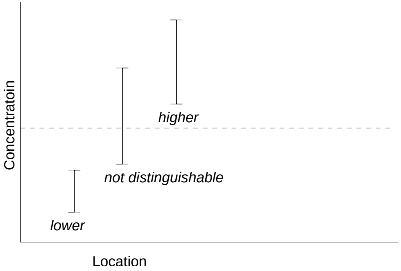
But:

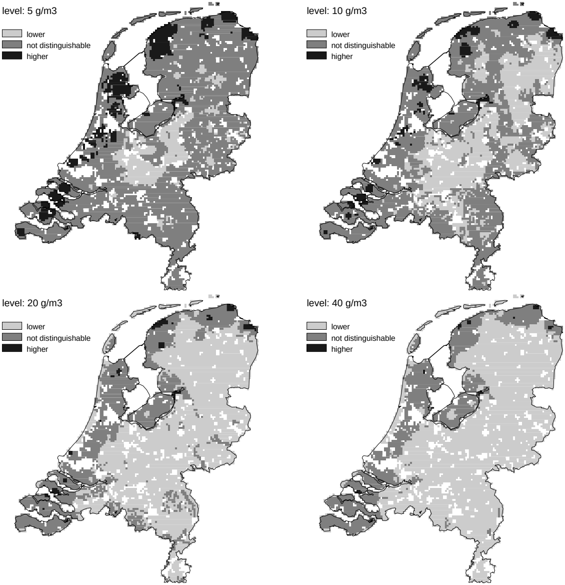
- "probabilities" may (and will) be outside $[0, 1]$
- it ignores whether $Z(s_i)$ is 501 or 5001

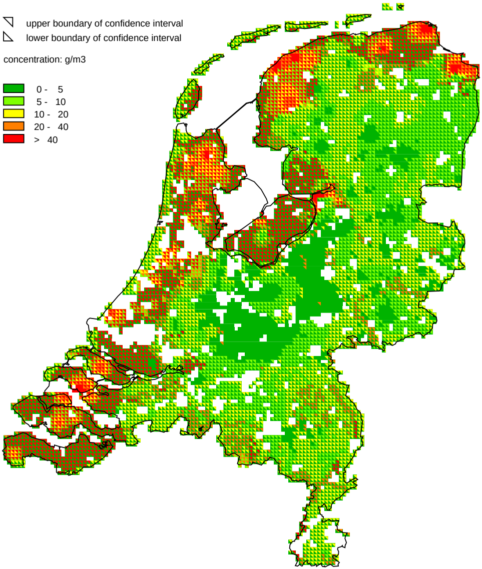
Maps with 95% confidence intervals

Usually, decision (smaller/larger) is done on the basis of (e.g. 95%) confidence interval:

- c.i. above threshold: larger
- c.i. below threshold: smaller
- else: undecided (“*not distinguishable*”)




level: 5 g/m3
lower
not distinguishable
higher
level: 10 g/m3
lower
not distinguishable
higher
level: 20 g/m3
lower
not distinguishable
higher
level: 40 g/m3
lower
not distinguishable
higher


◹ upper boundary of confidence interval
◺ lower boundary of confidence interval
concentration: g/m3
🟩 0 - 5
🟩 5 - 10
🟨 10 - 20
🟧 20 - 40
🟥 > 40

How to “compute” these?

```
p = out$p500
f = ifelse(p < .025, "lower", ifelse(p < .975, "not dist.", "higher"))
out$ci500 = factor(f, levels = c("lower", "not dist.", "higher"))
spplot(out["ci500"], col.regions = c("green", "grey", "red"))
```

Alternative: aguila (demo)

Conditional simulation

What is conditional simulation?

- it draws realizations instead of expected value, meaning that each sample is different
- the mean over all realizations equals the kriging mean, at each location
- the variance over all realizations equals the kriging variance, at each location
- the spatial correlation of a realization is (approximately) equal to the spatial correlation of $Z(s)$

When is conditional simulation needed?

- Simple answer: when the kriging + kriging variance map are not sufficient.
- What do kriging + kriging variance map not represent? Spatial correlation between interpolation errors. If two interpolation locations are nearby, their respective interpolation errors will be similar.
- When is this needed? An example (already mentioned) is block kriging–the block kriging mean equals the mean of point kriging values within the block, but the block kriging variance does not equal the variance of the point kriged values. However, block kriging is done differently.

When is CS needed?

- for non-linear spatial aggregation, e.g. block quantiles, block median, fraction of points within a block above a threshold
- for more complex spatial interaction: transport, flow, diffusion, ...
- when the interpolated map serves as input to a model with spatial interaction.

Some statistical methods always yield simulations instead of predictions; think of Markov Chain Mont Carlo, MCMC, in Bayesian computation.

Deterministic spatial dynamic models

Deterministic models are based on the assumption that the processes governing change are known. Given that knowledge, we need

- the state of a system (initial conditions), and
- the characteristics at boundaries of the system (boundary conditions): what are the sources and sinks, when does what escape or enter the modelled area.

for a (perfect) prediction of the changes over time, and in space.

Let us look at an example: air quality (fine particles, PM10).

Model domain

For a model, we need a model domain, which is the spatial area and temporal period over which we will define processes and conditions. This could be e.g. Western Europe, 1980-2010, or NRW, 2000-2005, or the area not further than 100 m away from the crossing Weseler Strasse-Bonhoeffer Strasse, Jan 8, 2008, 0:00-24:00. It should be exactly quantifiable/traceable.

Initial conditions

The initial conditions usually describe the variable of interest (the concentration field) at the highest possible level of detail. In our case this is the PM10 concentration field at the start of the modelling period.

As this is a continuous field, we need some way to describe this and usually the spatial domain is discretized into model usually square, rectuangular or triangular model elements.

This discretization should match the level of detail (i) at which we know initial conditions and (ii) at which we want to model features. As an example: if we want to quantify the effect of individual car fumes, spatial elements of 10 cm–1 m may work; if we want to describe the effect of a complete streets something of 10m–100m seems more appropriate. Smaller elements and time steps mean more memory and CPU time requirements.

Initial conditions

The initial conditions usually describe the variable of interest (the concentration field) at the highest possible level of detail. In our case this is the PM10 concentration field at the start of the modelling period.

As this is a continuous field, we need some way to describe this and usually the spatial domain is discretized into model usually square, rectuangular or triangular model elements.

This discretization should match the level of detail (i) at which we know initial conditions and (ii) at which we want to model features. As an example: if we want to quantify the effect of individual car fumes, spatial elements of 10 cm–1 m may work; if we want to describe the effect of a complete streets something of 10m–100m seems more appropriate. Smaller elements and time steps mean more memory and CPU time requirements.

Initial conditions

The initial conditions usually describe the variable of interest (the concentration field) at the highest possible level of detail. In our case this is the PM10 concentration field at the start of the modelling period.
As this is a continuous field, we need some way to describe this and usually the spatial domain is discretized into model usually square, rectuangular or triangular model elements.
This discretization should match the level of detail (i) at which we know initial conditions and (ii) at which we want to model features. As an example: if we want to quantify the effect of individual car fumes, spatial elements of 10 cm–1 m may work; if we want to describe the effect of a complete streets something of 10m–100m seems more appropriate. Smaller elements and time steps mean more memory and CPU time requirements.

Initial conditions-2

If we don't know initial conditions exactly, we may put the starting point of the modelling domain further back in the past, and hope that the effect of approximation will damp out as we model. (This assumes we get the boundary conditions and processes right.)

Boundary conditions

PM10 comes and goes. Sources are (i) emissions inside the model domain (cars, households, industry), and (ii) material that enters the model domain by movement of air bodies, but emitted elsewhere. We need these source terms (points, lines or fields) in space, and over time.
Sinks are mostly air that moves out of the model domain, and wash out (rain), dry deposition (your grandmother's white sheets turning black), and ... inhalation. These terms are also needed, quantitatively.

Processes

Particles move mostly for two or three reasons: by large-scale movement of air (wind), by medium/small-scale movement of air (turbulence, dispersion) and by themselves (diffusion; think Brownian motian of a single particle in a gas).
As an example, take a look at the LOTOS-EUROS model (`http://www.lotos-euros.nl/`) model documentation.
As you can read in the *model formulation and domain*, the model uses external modelling results (interpolation or mechanistic modelling) to get the atmospheric driving forces (height mixing layer, wind fields), e.g. from FUB and ECMWF (`http://www.ecmwf.int/`).
Basically, the model *code* (i) reads a lot of initial and boundary data, (ii) solves the differential equations and (iii) writes out everything that is of interest, such as the space-time concentration fields.

Solving differential equations

The partial differential equation solved,

$$\frac{\delta C}{\delta t} + U\frac{\delta C}{\delta x} + V\frac{\delta C}{\delta y} + W\frac{\delta C}{\delta z}$$

$$= \frac{\delta}{\delta x}(K_h\frac{\delta C}{\delta x}) + \frac{\delta}{\delta y}(K_h\frac{\delta C}{\delta y}) + \frac{\delta}{\delta z}(K_z\frac{\delta C}{\delta z}) + E + R + Q - D - W$$

needs to be discretized in space and time. Spatial grid size is 0.5°long × 0.25°lat (meaning that grid cells do not have constant area), and time step is 1h.

Solving PDE's

The simples method, *finite difference*, uses a regular mesh size, Δx. In one dimension the first derivative uses one of the three approximations (backward, forward, centered):

$$\frac{\delta u}{\delta x}(j\Delta x) \approx \frac{u_j - u_{j-1}}{\Delta x}$$

$$\frac{\delta u}{\delta x}(j\Delta x) \approx \frac{u_{j+1} - u_j}{\Delta x}$$

$$\frac{\delta u}{\delta x}(j\Delta x) \approx \frac{u_{j+1} - u_{j-1}}{2\Delta x}$$

and for the second order derivative

$$\frac{\delta^2 u}{\delta x^2}(j\Delta x) \approx \frac{u_{j+1} - 2u_j + u_{j-1}}{(\Delta x)^2}$$

Diffusion equations, 1-D

Diffusion happens in space-time. Using a mesh in space-time, we can write $u(j\Delta x, n\Delta t) \approx u_j^n$ with n a superscript, not power. We can approximate

$$\frac{\delta u}{\delta t}(j\Delta x, n\Delta t) \approx \frac{u_j^{n+1} - u_j^n}{\Delta t}$$

$$\frac{\delta u}{\delta x}(j\Delta x, n\Delta t) \approx \frac{u_{j+1}^n - u_j^n}{\Delta x}$$

PDE:

$$\frac{\delta u}{\delta t} = \frac{\delta^2 u}{\delta x^2}, \text{ with } u(x,0) = \phi(x)$$

Using forward difference for t and centered for x, the corresponding finite difference equation that approximates it is:

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2}.$$

Forward/backward, explicit/implicit

Solving

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2}.$$

is trivial, as $n + 1$ is only in the LHS. This means that for each x we can solve the equation explicitly, where we start is not important. They require, for stable solutions, that $\frac{\Delta t}{(\Delta x)^2} \leq \frac{1}{2}$. See examples. If the equation were instead

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{(\Delta x)^2}.$$

then we have the unknown u^{n+1} both left and right of the equal sign. This requires the solution of a (sparse) set of coupled linear equations, and this solution is called *implicit*. It pays off: the solutions are stable, and larger time steps can be chosen (provided of course, that change is close to linear over the time step).

Calibrating deterministic models

Models based on partial differential equations have parameters; think of diffusion parameters, source and sink terms (boundary conditions), and initial conditions. These need to be "filled in", somehow.

Given that observation data on the model outcome are available, one way to fill these in is to search for values such that the model predictions best fit the data. We have seen methods for this; there is a long list of further, possibly more advanced or efficient methods for finding optimal parameter values, both in a deterministic ("optimum") sense, and in a stochastic ("distribution") sense.

Also, choosing optimality criterium (least squares? least absolute differences? combined criteria over multiple variables?)

Difficulties in calibration

Problems that may occur with calibrating models are numerous. One problem may be that the parameter we tune (optimize) is not constant over space or time, but varies. This means that there instead of one single value, there may be numerous. Their number may outnumber the observations, and in that case there is little hope in finding realistic values.

Another problem is that we may tune a parameter and get a better fit, but that in reality we turned the wrong "button", meaning we get a better fit *for the wrong reason*. This may have disasterous effects when using this "optimized" model in a prediction setting (such as future forecasting, or scenario evaluation).

Automatic codes exist (e.g. "PEST", or R's `optim`) that optimize models, irrespective what the model is or does.

More difficulties

Deterministic models use a temporal and spatial discretization. This is a balance between CPU and memory costs, and the ability to fill the discrete elements sensibly. Processes need to be “lumped”, meaning that they cannot be taken into account because of the grid cell size (think of convection above a forest, or a thunder storm, when grid cell size is 50 km, and/or time step a day). Choosing a finer resolution, the parameters, processes, boundary and initial values need to be filled in with much more resolution (precision), and need disaggregation – e.g. a country total emission may need to be assigned to 1 km × 1 km grid cells.

Dynamic parameter updating schemes

A probabilistic setting of a deterministic model is that of the *Kalman filter*. This algorithm assumes measurements are a combination of a true state and a measurement noise. Each componentn has its particular error structure, expressed by a mean and covariance matrix.

For each new time step, the model predicts a new state, the observations are compared to that new state, and the model errors are used to adjust (improve) the model before predicting the next step.

Kalman filters are used a lot to optimize deterministic models, and come nowadays in many flavours, e.g. depending on whether the model is linear or not, and whether it is used forward in time, or in real-time.

Simplified difference equation-type models

Often, the differential equations are simplified very crudely to the state where only mass is preserved, but the solution no longer approximates the true differential equation. Think of simple bucket-type models in hydrology, where water bodies are buckets and soils are like sponges: a soil grid cell drains always with an exponential decrease; a soil and water body grid cells drain "instantly" when their maximum capacity is exceeded, with the amount it is exceeded.

Despite the simplifications, these models can be more useful than attempts to solve the full differential equation, because their data demand can be more realistically met.

This is not a plea against the use of any model

On the contrary, elemenents from physics (such as preservation of energy and mass) are better than anything else. It is rather a warning, that putting them blindly before anything else is dangerous.

In any case one should acknowledge the limited availability of data available for calibration, and the limitations of our ability to truthfully represent real-world systems for the world around us.

The risk of inapt use (“misuse”) is not larger for deterministic models than for stochastic models. But we do tend to rely on them in more risky situations (scenario’s, future).

Which approach to choose?

When we have the information (process knowledge and necessary inputs and boundaries) available, and where the data of the variable of interest are sparse or missing (scenario's, future), a deterministic modelling always needs to be the core of sound scientific practice.

In the luxuous position where both information for the process is available and measurements are abundant, stochastic modelling of the observations (think *ordinary kriging*) may be sufficient as well.

Stationary and non-stationary spatio-temporal models

In the context of differential equation type models, the word *stationarity* usually refers to the temporal behaviour: a stationary model solves the equation for a situation where there is no change over time. This means there is balance between inputs and outputs. This solution is valid for as long as boundary conditions do not change. This is especially useful for stable systems without large short-term fluctuations, where we are interested in some system change (scenario), e.g. groundwater hydrology.
The non-stationary case generalizes this to the cases where there is temporal change. The equation itself (convection-dispsersion) models a non-stationary situation, and represents a dynamic spatial system by its nature.

Dynamic stochastic models

In the kriging model, $y(s) = m + e(s)$, there is no reason to limit the spatial index s to two-dimensional space; it can be one-dimensional (in which case we would apply geostatitics to time series data, and would ignore the possibility of markov-type assumptions), two-dimensional (think of the exercises), three-dimensional (think mining applications, or water bodies), or space-time ($s = (x, y, t)$, or even $s = (x, y, z, t)$). In space-time indexes, one should *always* address the issue that time units principally differ from space units, and quantify how fast autocorrelation decreases in time, and in space. The case where autocorrelation decreases faster in some directions than in others is called *anisotropy*, when it decreases equally fast, direction-independent, it is called *isotropy*.

Anisotropy can be seen as a *space transform*.

Dynamic stochastic models

In the kriging model, $y(s) = m + e(s)$, there is no reason to limit the spatial index s to two-dimensional space; it can be one-dimensional (in which case we would apply geostatitics to time series data, and would ignore the possibility of markov-type assumptions), two-dimensional (think of the exercises), three-dimensional (think mining applications, or water bodies), or space-time ($s = (x, y, t)$, or even $s = (x, y, z, t)$). In space-time indexes, one should *always* address the issue that time units principally differ from space units, and quantify how fast autocorrelation decreases in time, and in space. The case where autocorrelation decreases faster in some directions than in others is called *anisotropy*, when it decreases equally fast, direction-independent, it is called *isotropy*.

Anisotropy can be seen as a *space transform*.

Dynamic stochastic models: prediction

Space-time autocorrelations need be anisotropic by the nature of space and time.
Given variance(s) and space-time autocorrelation structure, the kriging equations do not change from those given above.

Detecting and modelling anisotropies

In space: instead of computing semivariances over all directions, we can limit the inclusion of point pairs for the sample semivariogram to those that are aligned North-South, or e.g. those that are within a direction tolerance of $\pm 22^o$ from North-South. Next, we do the same for E-W, and for NE-SW and NW-SE.

In space-time: often, sensor locations do not change over time so it is easy to construct autocorrelograms (or semivariograms) over time only from a single sensor. Next, we could average these over space to get the autocorrelation in time only. For time slices, we can estimate autocorrelation in space only.

Through all the directional semivariograms a single (possibly anisotropic) model needs to be fitted. The simples is a single model for which the range parameter changes with direction, e.g. using an ellipse (*geometric anisotropy*). More complex is a model where (also) the variance (sill) depends on direction (*zonal anisotropy*).

Detecting and modelling anisotropies

In space: instead of computing semivariances over all directions, we can limit the inclusion of point pairs for the sample semivariogram to those that are aligned North-South, or e.g. those that are within a direction tolerance of $\pm 22^o$ from North-South. Next, we do the same for E-W, and for NE-SW and NW-SE.

In space-time: often, sensor locations do not change over time so it is easy to construct autocorrelograms (or semivariograms) over time only from a single sensor. Next, we could average these over space to get the autocorrelation in time only. For time slices, we can estimate autocorrelation in space only.

Through all the directional semivariograms a single (possibly anisotropic) model needs to be fitted. The simples is a single model for which the range parameter changes with direction, e.g. using an ellipse (*geometric anisotropy*). More complex is a model where (also) the variance (sill) depends on direction (*zonal anisotropy*).

Detecting and modelling anisotropies

In space: instead of computing semivariances over all directions, we can limit the inclusion of point pairs for the sample semivariogram to those that are aligned North-South, or e.g. those that are within a direction tolerance of $\pm 22^o$ from North-South. Next, we do the same for E-W, and for NE-SW and NW-SE.

In space-time: often, sensor locations do not change over time so it is easy to construct autocorrelograms (or semivariograms) over time only from a single sensor. Next, we could average these over space to get the autocorrelation in time only. For time slices, we can estimate autocorrelation in space only.

Through all the directional semivariograms a single (possibly anisotropic) model needs to be fitted. The simples is a single model for which the range parameter changes with direction, e.g. using an ellipse (*geometric anisotropy*). More complex is a model where (also) the variance (sill) depends on direction (*zonal anisotropy*).

Differential models with stochastic paramaters or mechanisms

The story (complexity) does not end here. An example of combined stochastic and deterministic modelling is a stochastic differential equation. It is a differential equation that contains one or more random variables rather than fixed.
An example is a diffusion equation that contains random walk as its elementary mechanism. A solution can only be described in terms of a probability density function, as the "true" outcome will always be unpredictable (because it is subject to chance).

Additive or multiplicative errors?

In the exercises on the meuse data set, we have worked with the model

$$y = X\beta + e$$

where y are the log-transformed data. This is equivalent to the model where the errors are proportional, say $z = X\beta \cdot \tilde{e}$ with $\tilde{e}$ an error with mean 1:

$$\log(z) = \log(X\beta\tilde{e}) = \log(X\beta) + \log(\tilde{e})$$

This means that if $\tilde{e}$ and e have constant variance, on the log-scale the variance of z is proportional to the mean.
Deciding whether on the observation scale or on a log-transformed scale certain properties (constant variance, stationarity) hold, is the responsibility of the modeller (i.e., you).

General transforms: Box-Cox

Not transforming (or any linear transform: shift and/or multiplication) and log-transforming are only two options. The Box-Cox family of continuous transformations generalizes these, and includes all power transforms:

$$f(y,\lambda) = \begin{cases} (y^\lambda - 1)/\lambda & \text{if } \lambda \neq 0, \\ \ln(y) & \text{if } \lambda = 0 \end{cases}$$

Note that y needs to be positive if $\lambda = 0$, or else non-negative (except for negative odd integer values of λ).

Example Box-Cox

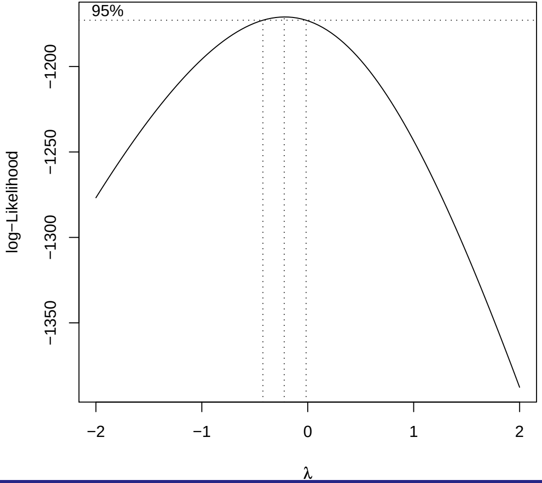
Which value for λ is optimal for the linear model

$$f(\text{zinc}, \lambda) = \beta_0 + \beta_1 \sqrt{\text{Dist}} + e,$$

such that e follows a distribution as close as possible to normal?
The following R code solves this:

```
> library(MASS) # Modern Applied Statistics with S, Venable
> library(sp)
> data(meuse)
> out = boxcox(lm(zinc~sqrt(dist), meuse))
> out$x[which(out$y == max(out$y))]
[1] -0.2222222
```

Example Box-Cox – plot



More warnings

Transformation may not solve all your problems. It is for example well-known that for count data we need to log-transform to get normal residuals, and a square-root transform to get residuals with a constant variance.

Modelling: stochastic, deterministic, or combined?

This is a general research question with no single answer.
In linear regression, the data are decomposed as

$$y = X\beta + e$$

with y the observations (random), e a residual (random) and $X\beta$ the trend (deterministic, fixed). It is only lack of data that cause the need for inference (testing) on e.g. $\hat{\beta} - \beta$.
In essence, this means that we split variability in something that we understand ($X\beta$) and something we don't understand e, because it varies at random.
One could argue that understanding is always better than randomness; we should only use random models after we run out of means to understand.