Norsk Regnesentral
NORWEGIAN COMPUTING CENTER

www.nr.no

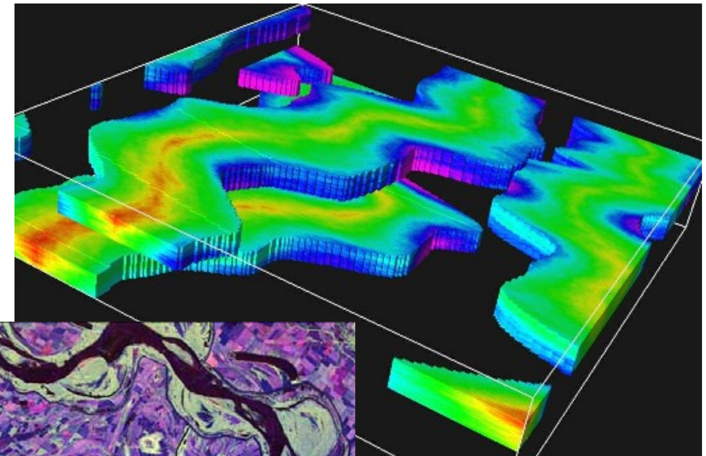
STK 4050 Stochastic Simulation – Some Practical Applications and Problems

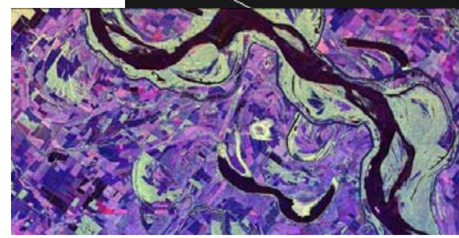
Petter Abrahamsen

Content

- Background – what we do
- Why do we simulate?
- Example – oil in place
- Software demo of RMS (commercial software)
- Simulation vs. conditional simulation
  - When does conditioning work?
- Example: Use spill point information
- The Alvheim decision: Big or small boat?
- BREAK
- Gaussian random fields
  - Simulation techniques
- Categorical random fields (e.g. rock types)
  - Object models
  - Truncated Gaussian
  - Indicator kriging
  - Markov random fields
  - Multipoint algorithms
- The Snesim (multipoint) algorithm
  - Idea and description
  - What works and what doesn't work?
  - Possible fix?
  - Discussion
- Simulation in practice – some closing remarks

What do we do at NR?

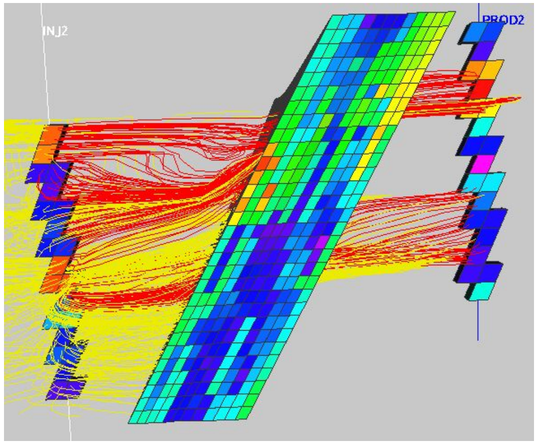




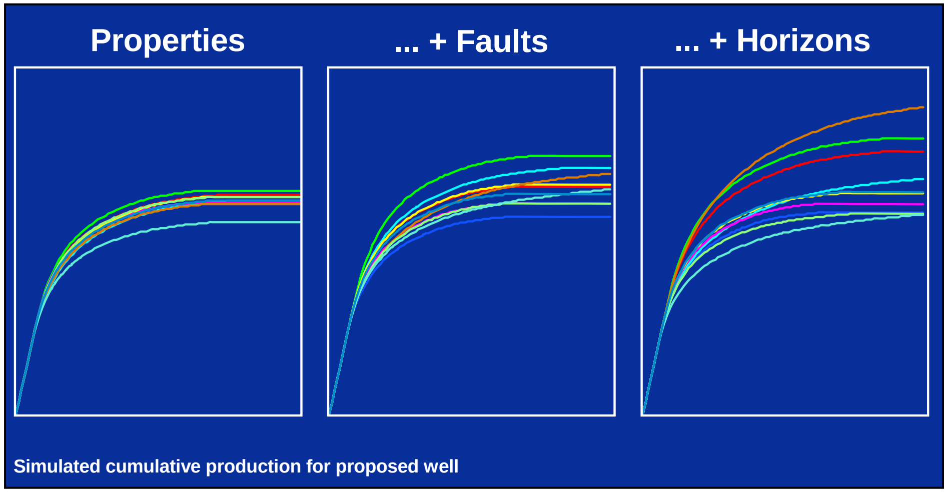
- ► Model geology and nature
  - ▪ Partly systematic (geological process)
  - ▪ Partly random (weather and climate changes)
- ► Spatial statistics
- ► High dimensional distributions
  - ▪ E.g. 200 x 200 x 200 = 8 000 000 cells
- ► Data integration – conditional simulation

Why do we simulate?

- Non-Gaussian distributions – math can be very difficult
- High dimension
- Non-linear relationships:
  $E[f(X)] \neq f(E[X])$, etc.

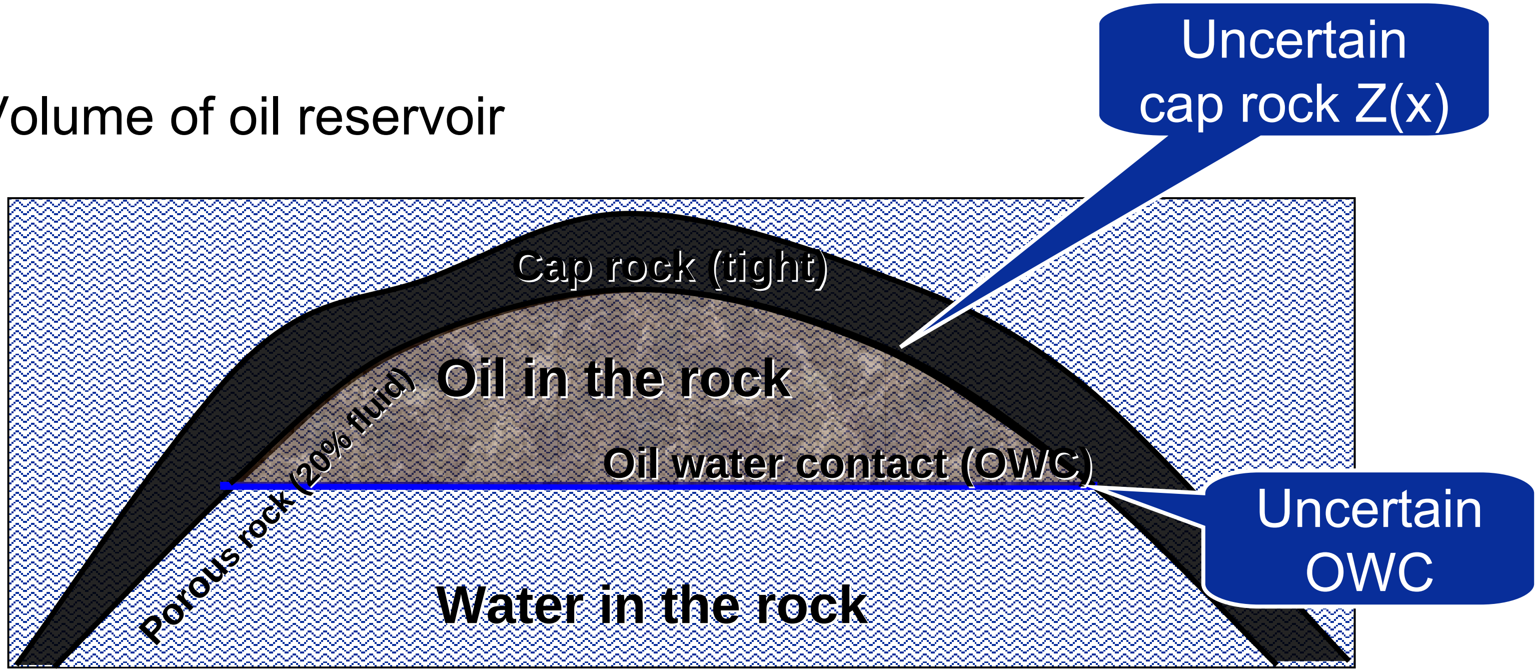


- Very flexible approach – can use *any* transformation *f*
- Often easy and intuitive to simulate – easy to communicate results



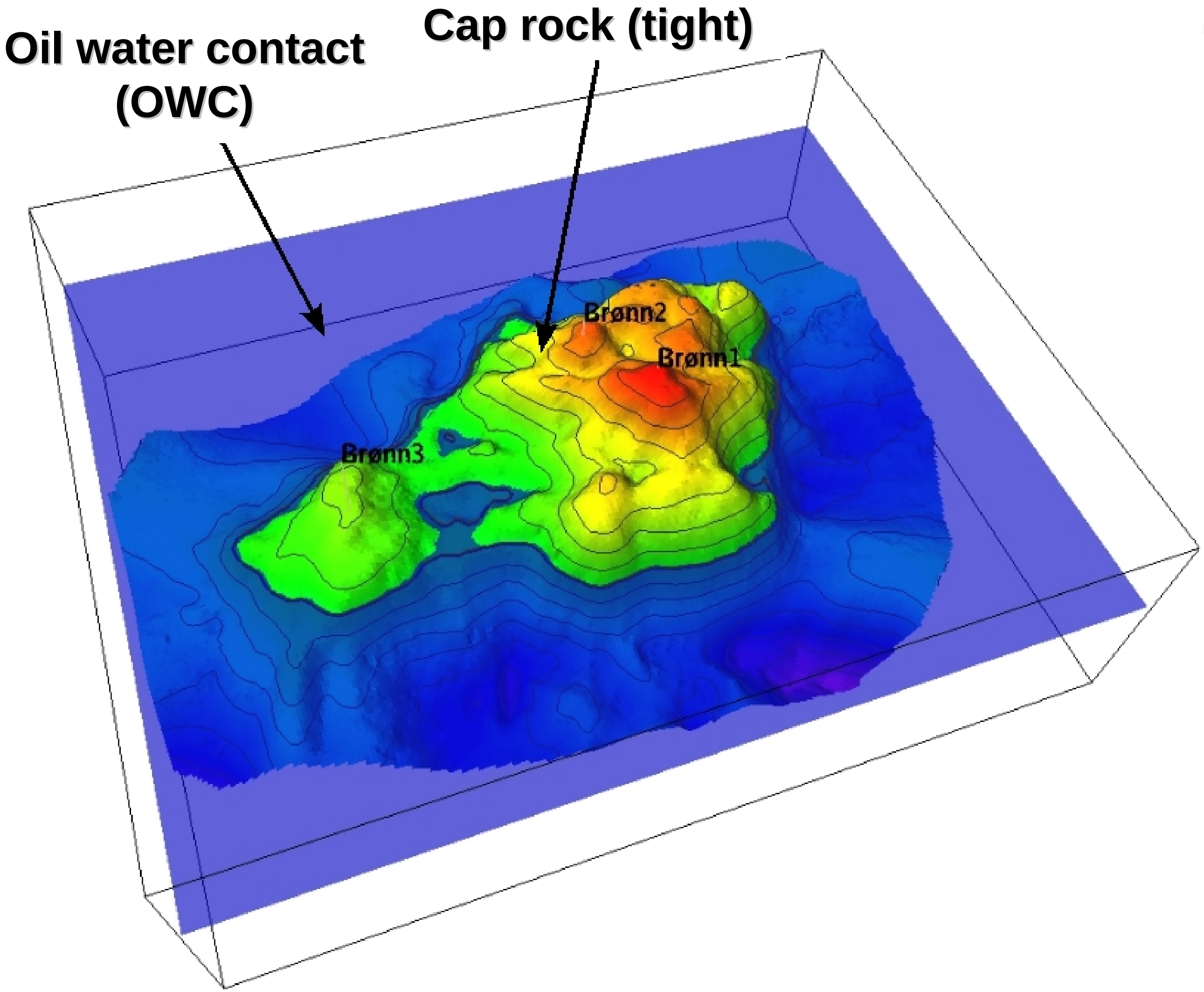
Example – oil in place

- Volume of oil reservoir



- *Volume* = $\int_D \max(0, Z(x) - OWC(x))\,dx$
- Assume
  - OWC ~ Known or e.g. Gaussian
  - Z(x) is a Gaussian random field
- When will *Volume* be Gaussian?

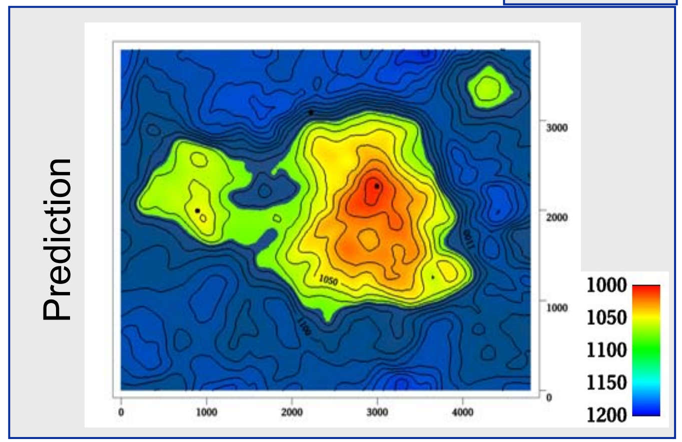
Expected OWC and cap rock

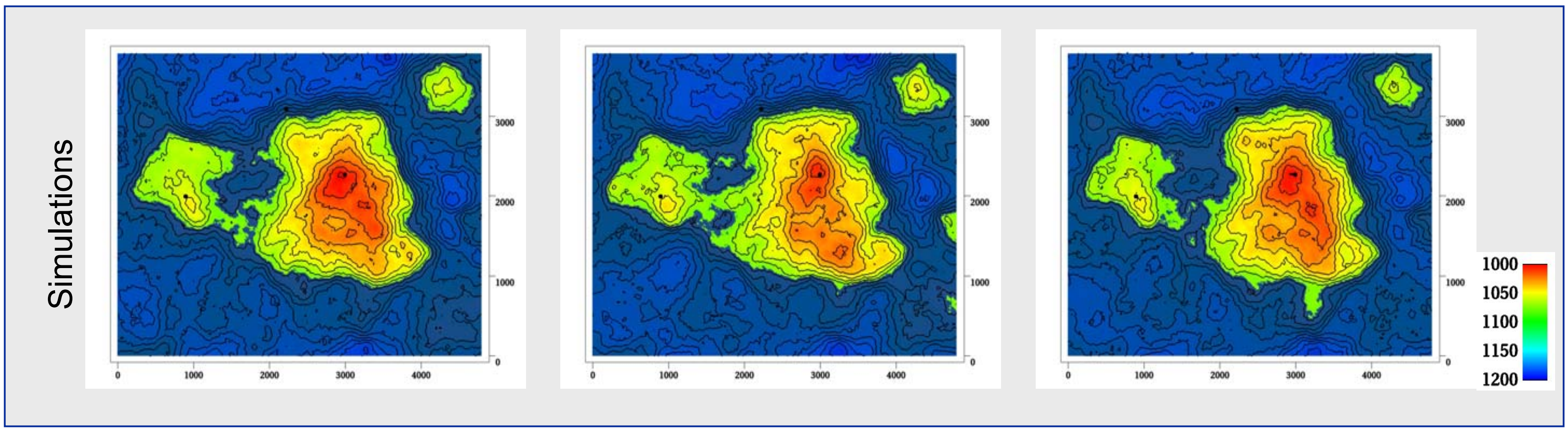


Why simulation?

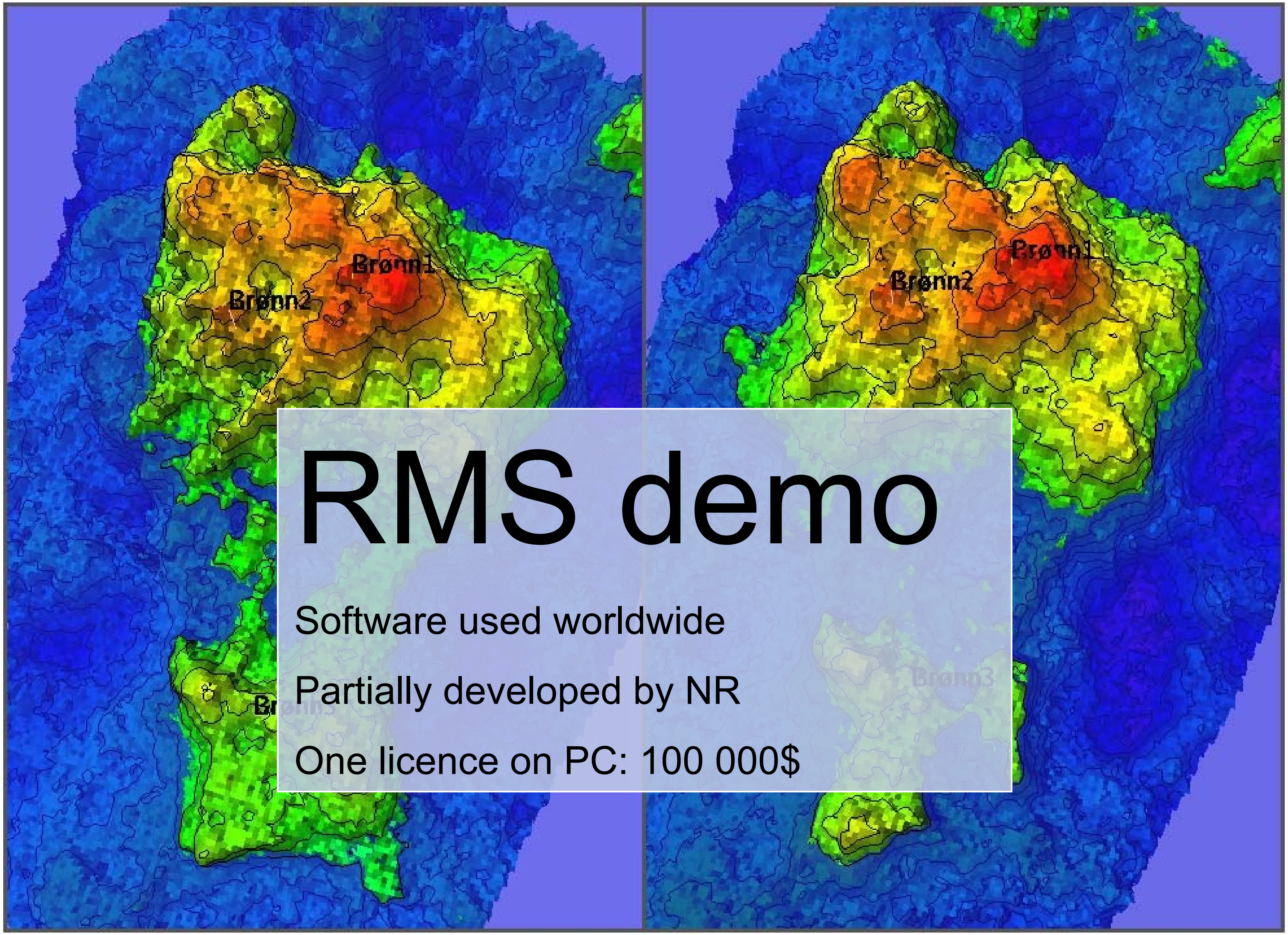
- Simulation is necessary to get non-linear properties correct:
  - Volume above oil water contact
  - Drainable area

OWC = 1080





Simulated cap rock



RMS demo

Software used worldwide

Partially developed by NR

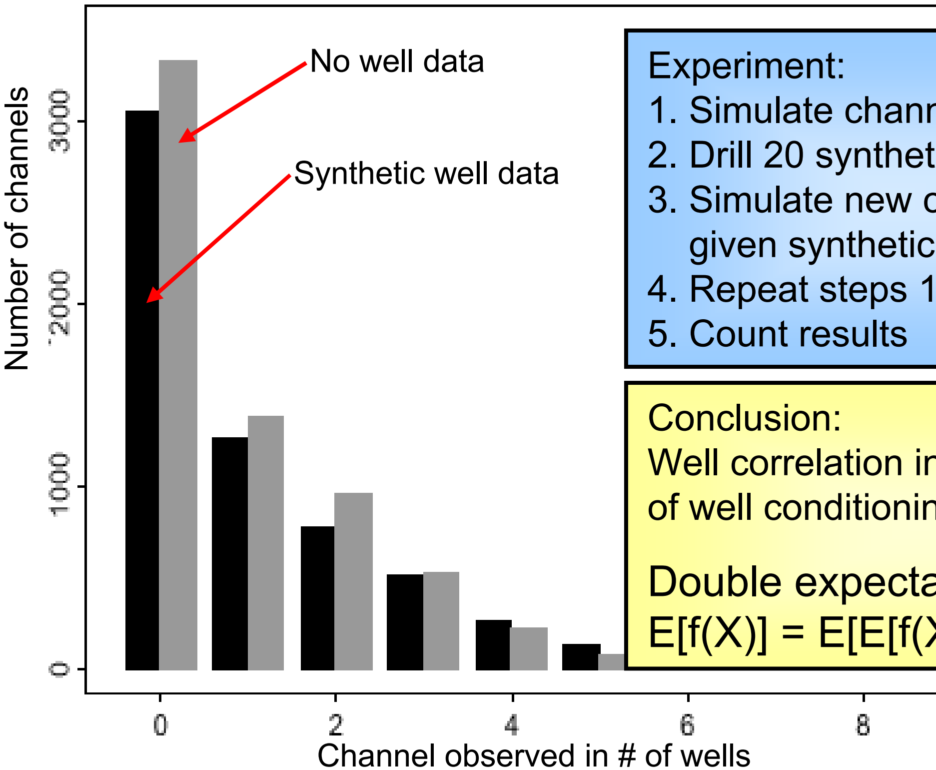
One licence on PC: 100 000$

Simulation vs. conditional simulation

- Want to draw from P(x|data) *not* from P(x)
  - (Often a Bayesian formulation)
- Rejection sampling:
  - Draw from P(x)
  - Reject if x in conflict with data
  - Usually extremely inefficient
- MCMC methods
  - Time consuming in high dimensional cases
  - Simulated annealing to obtain conditioning
- Direct sampling from P(x|data)
  - Requires partly analytical solution and efficient approximations

Consistency experiment

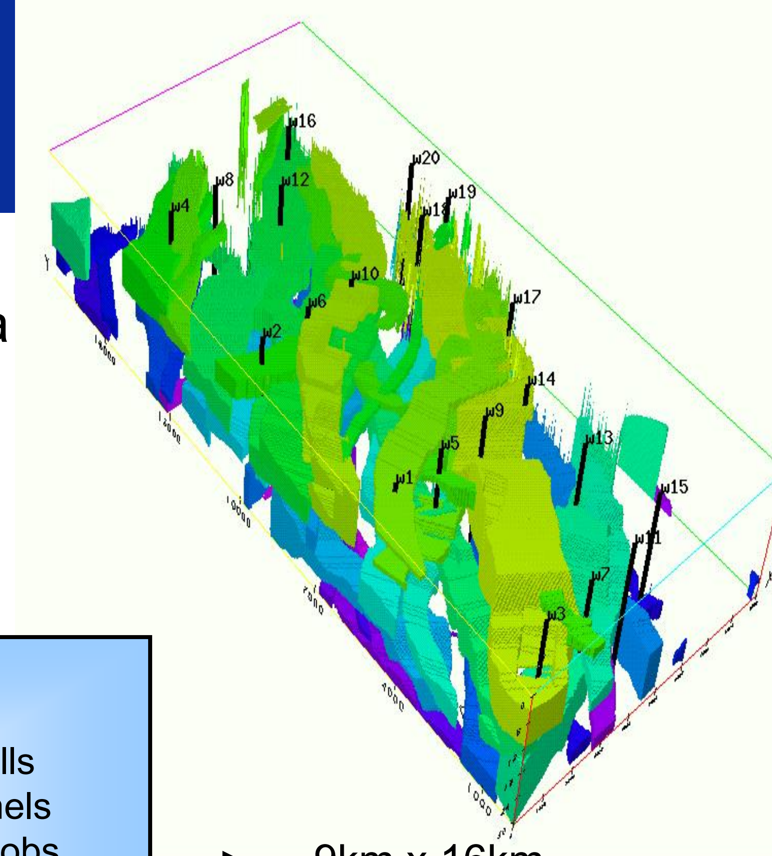
- Model behaviour independent of data
  - E.g. connectivity independent of well conditioning
  - Non-drilled areas have the same connectivity properties



Experiment:
1. Simulate channels
2. Drill 20 synthetic wells
3. Simulate new channels given synthetic well obs.
4. Repeat steps 1-3 100 times
5. Count results

Conclusion:
Well correlation independent of well conditioning.

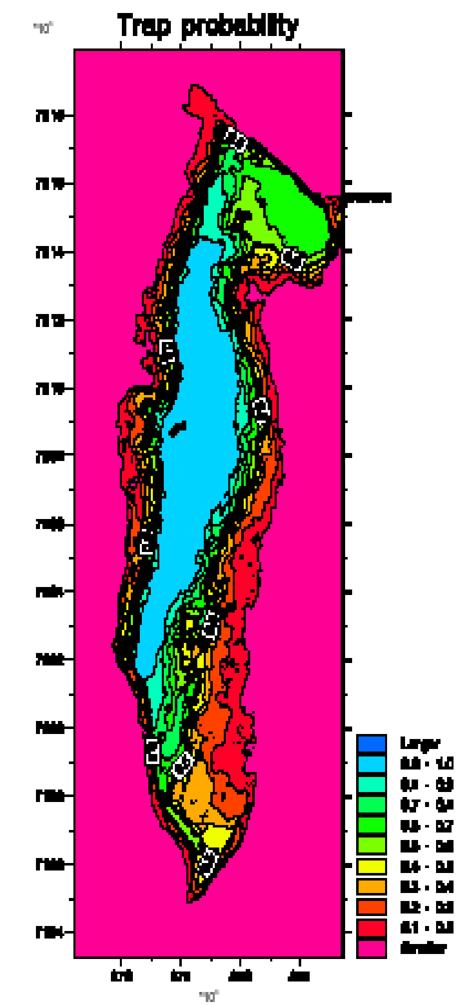
Double expectation correct:
$E[f(X)] = E[E[f(X|Data)]]$



- 9km x 16km
- 20 vertical wells
- Channel width:
  - ~ N(700m,$500^2 m^2$)
  - > 200m
- 40 - 60 channels in each realisation
- NG: 36% - 40%

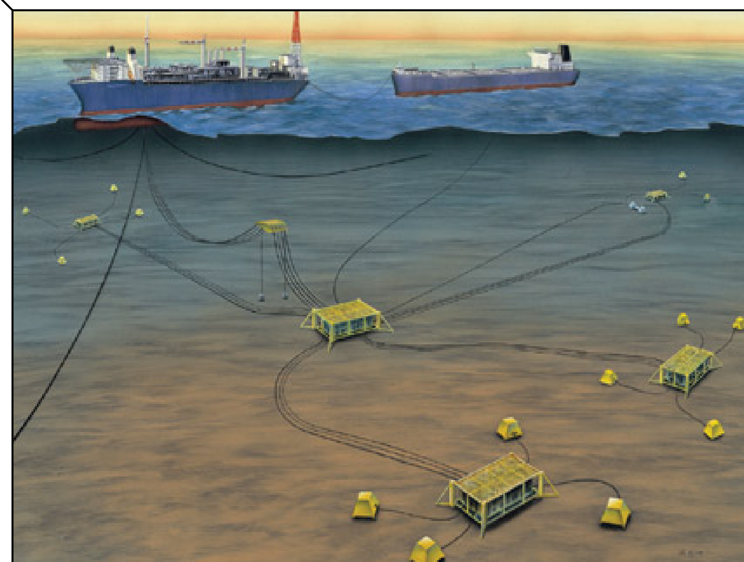
Example: Using spill-point information

- Illustrated by case-study from Norskehavet



The Alvheim decision

- Big or small boat?



Sampling Gaussian RF

- Consider $X(s) \sim$ GRF, $s \in R^n$
  $E[X(s)] = \mu(s)$, $\mathrm{Cov}(\mathbf{X}_1 | \mathbf{X}_2) = \Sigma_{12,}$
- Want to draw $\mathbf{X}_1 | \mathbf{X}_2 = \mathbf{x}_2$, ($\mathbf{X}_1$ typically a large lattice/grid)
- Recall:
  1. $\mathbf{X}_1 | \mathbf{X}_2 = \mathbf{x}_2 \sim N(\mu_1 + \Sigma_{12} \Sigma_{22}^{-1} (\mathbf{x}_2 - \mu_1), \Sigma_{11} - \Sigma_{12} \Sigma_{22}^{-1} \Sigma_{21})$
  2. $\mathbf{X} = \mu + \Sigma^{1/2} \varepsilon, \varepsilon \sim N(0, \mathbf{I})$
- Typical dimensions:
  $\dim(\mathbf{x}_1)$ = 100 000 – 10 000 000 (huge grid)
  $\dim(\mathbf{x}_2)$ = 10 – 10 000 (observations)

GRF simulation – possible strategies:

- Two step approach
  1. Unconditional simulation: $\mathbf{x}_1^s$
  2. Conditioning: $\mathbf{x}_1^s - \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2^s - \mathbf{x}_2)$
  - So how do we get $\mathbf{x}_1^s$?

- Sequential simulation:
  1. Draw $x(s_1)|\ \mathbf{X}_2=\mathbf{x}_2$
  2. Draw $x(s_2)|\ \mathbf{X}_2=\mathbf{x}_2,\ x(s_1)$

  n. Draw $x(s_n)|\ \mathbf{X}_2=\mathbf{x}_2,\ x(s_1),\ldots,\ x(s_{n-1})$

  - How do we cope with all that conditioning data?

Simulation method: Two step approach in several steps…

- Mean value

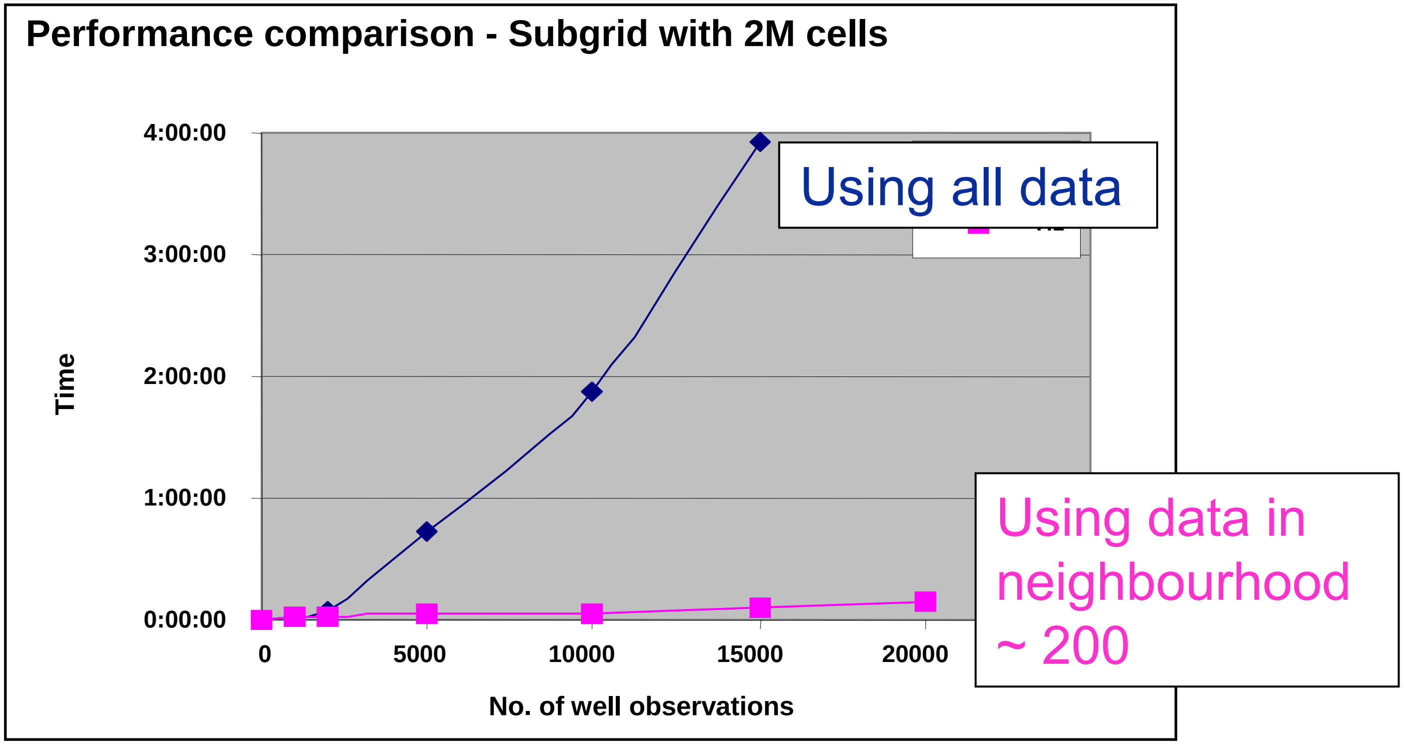
1. Simulate residual (using e.g. FFT algorithm)
2. Add mean and residual
3. Find difference between data and simulated field
4. Use simple kriging to interpolate this difference
5. Add interpolated difference to simulated field

FFT

$$x^s(\mathbf{s}) = \mu(\mathbf{s}) + r^s(\mathbf{s}) + \boldsymbol{\Sigma}_2'(\mathbf{s})\boldsymbol{\Sigma}_{22}^{-1}\left(\mathbf{X}_2 - \boldsymbol{\mu}^s - \mathbf{r}^s\right)$$

Do we need any approximations?

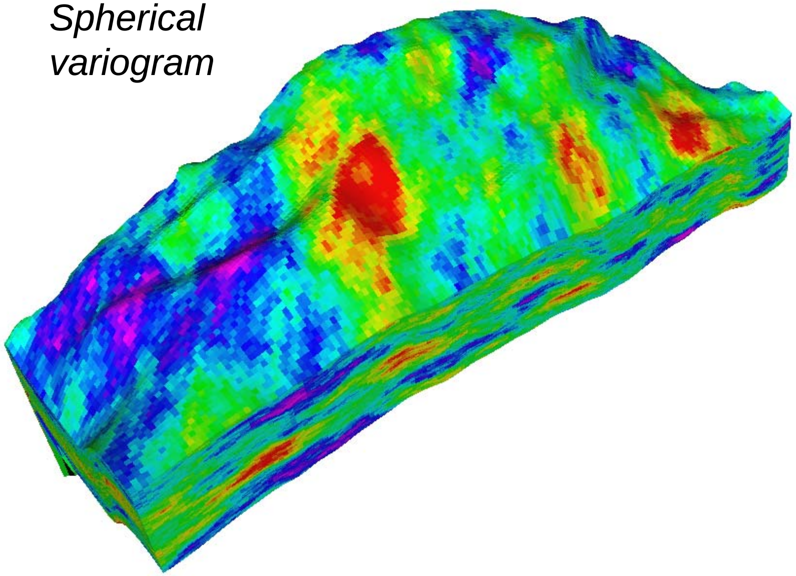
CPU usage



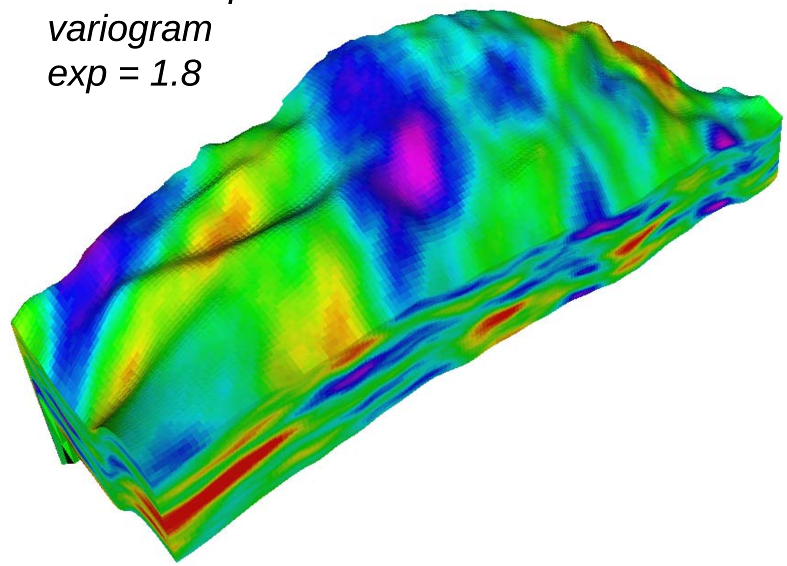
- FFT part is irrelevant – conditioning to data is the challenge

This is what it looks like:

Spherical variogram



General exponential variogram exp = 1.8



Sequential simulation

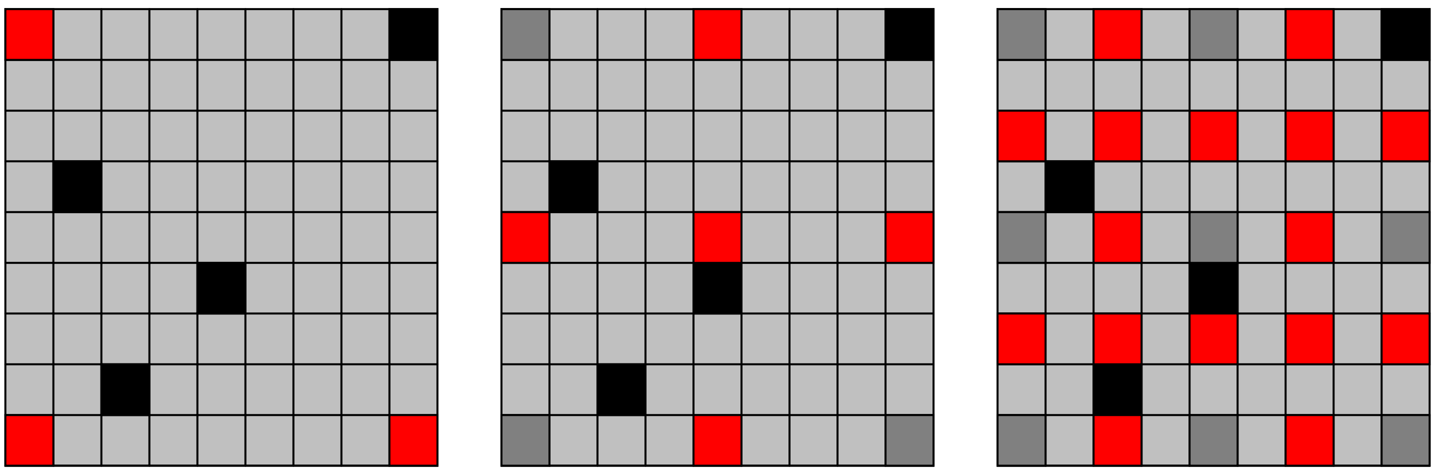
- Exact since

  $P(x_1,\ldots,x_n) = P(x_1)\cdot P(x_2|x_1)\cdot P(x_3|x_1,x_2)\cdot P(x_4|x_1,x_2,x_3)\cdots P(x_n|x_1,\ldots x_{n-1})$

- Necessary approximation: Only consider x's in a (small) neighborhood:

  $P(x_k|x_1,\ldots x_{k-1}) \approx P(x_k|\partial(x_k))$

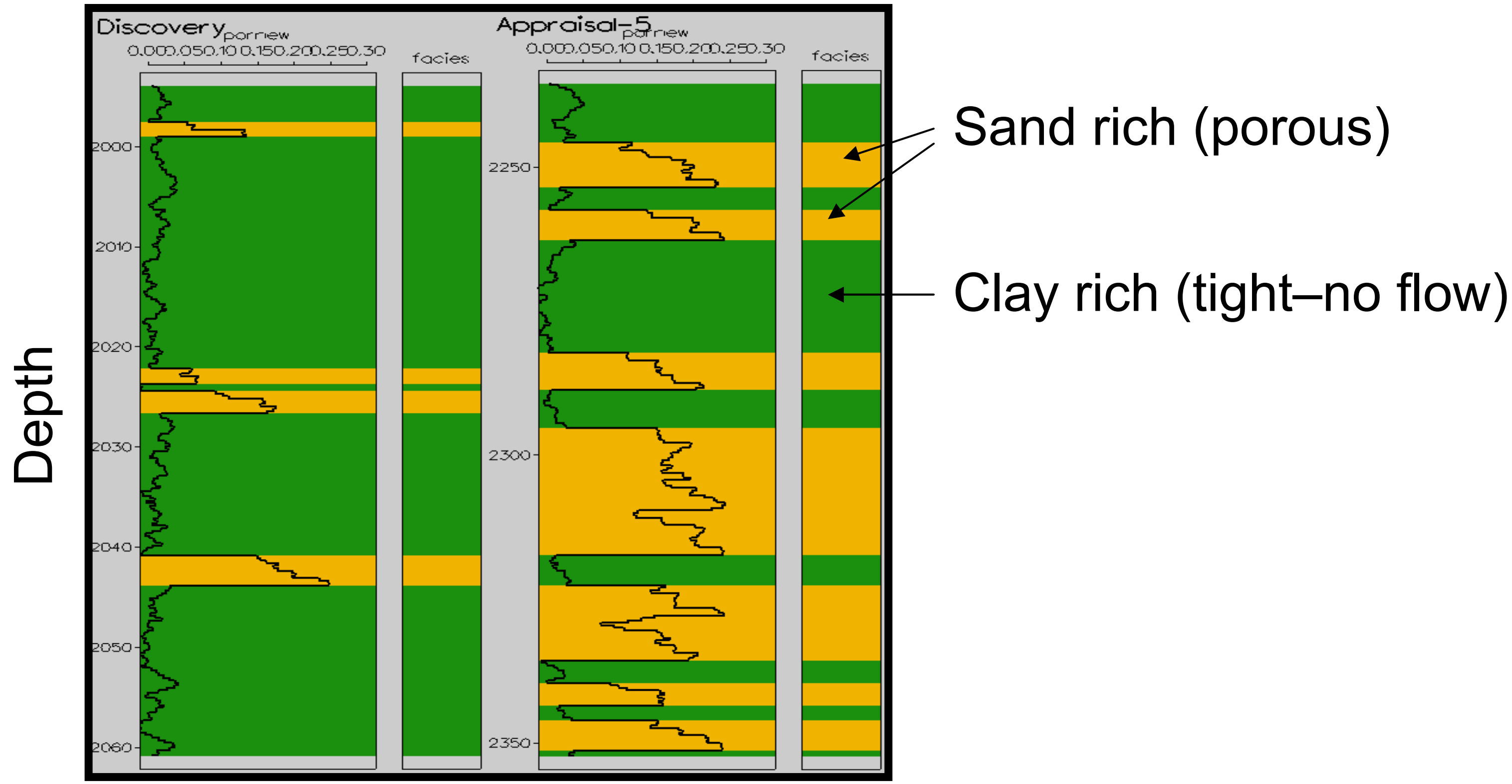
- Random path through grid follows a refinement scheme:



- Ensures good large-scale behavior

Categorical random variables

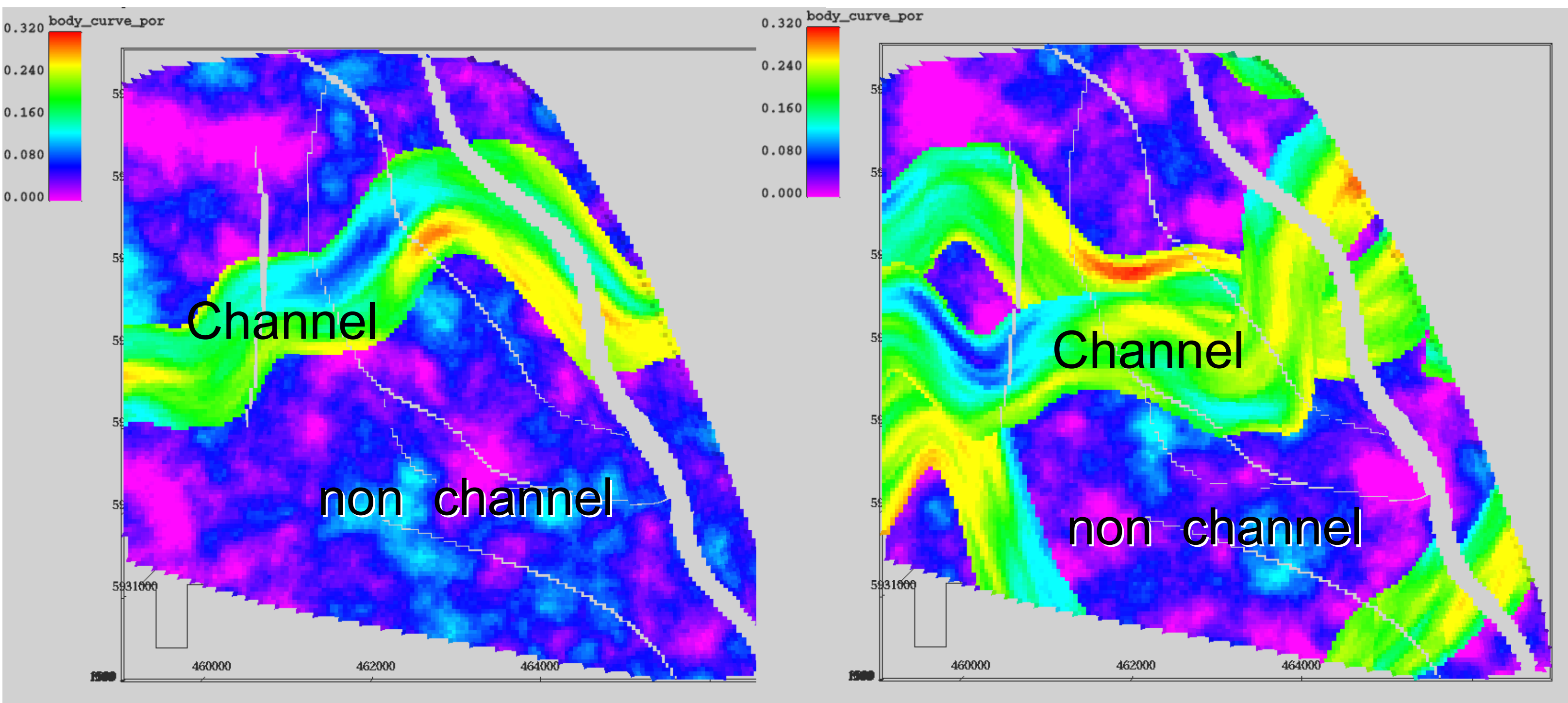
- We use it for classification of rock types



Porosity logs (percentage of open space)

Discrete random variables

- Here we see sand rich channels with high porosity

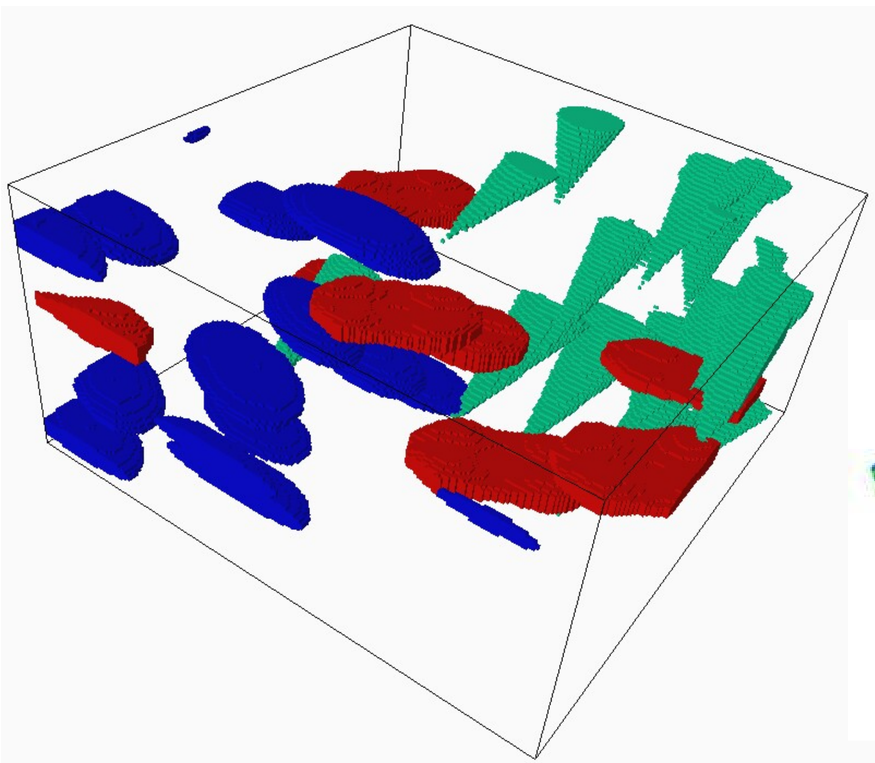


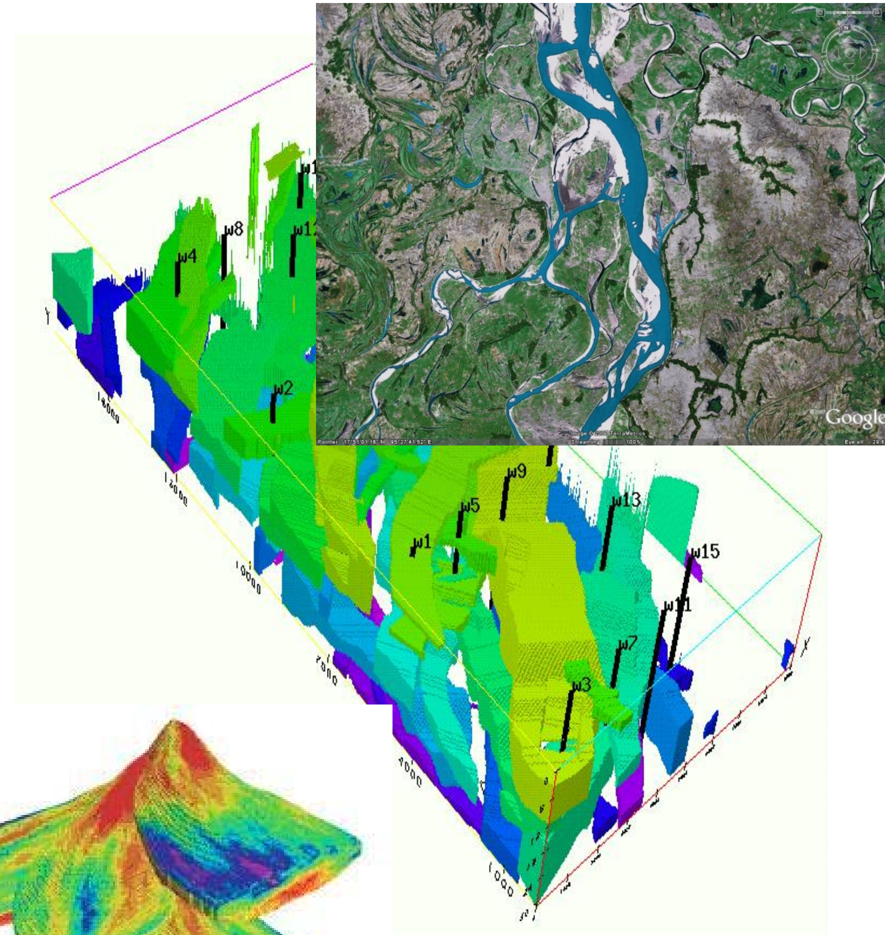
How do we simulate discrete patterns

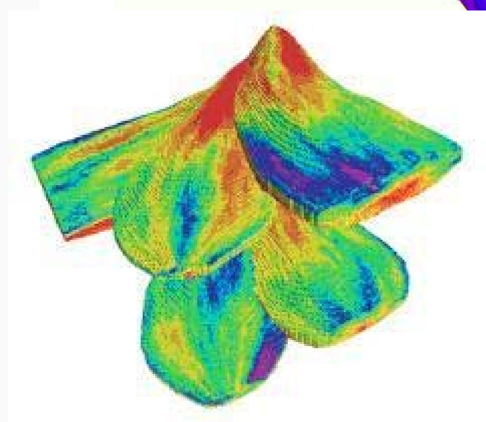
- Object models (marked point processes)
- Truncated Gaussian random fields
- Indicator kriging
- Markov random fields
- Multipoint algorithms

Object models

- Distinct geometries
  - Shape, size, etc.
- Challenge to condition to data

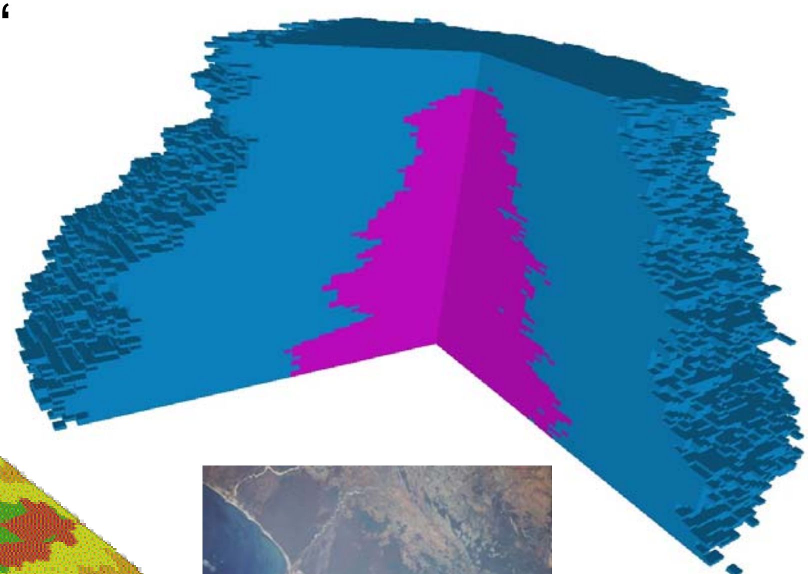


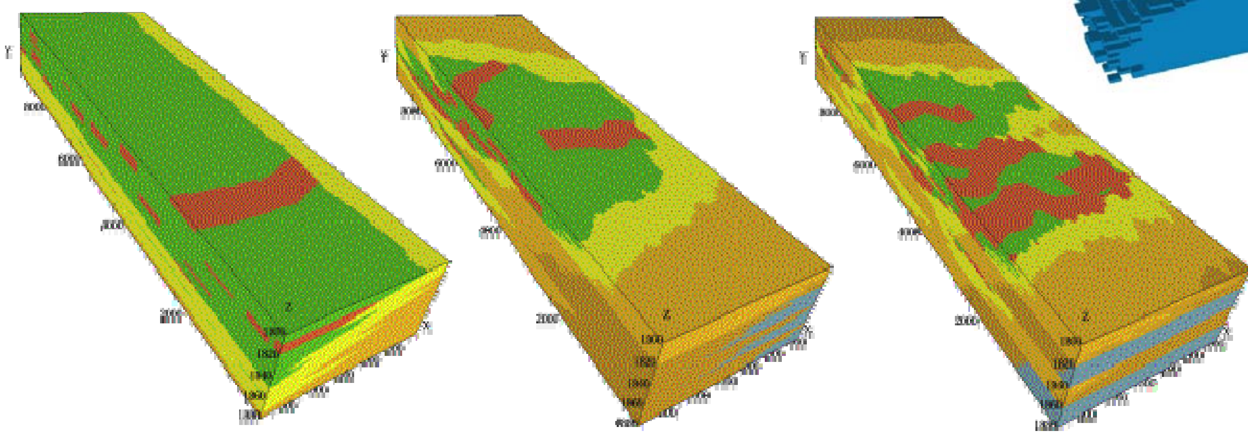




Truncated Gaussian random field

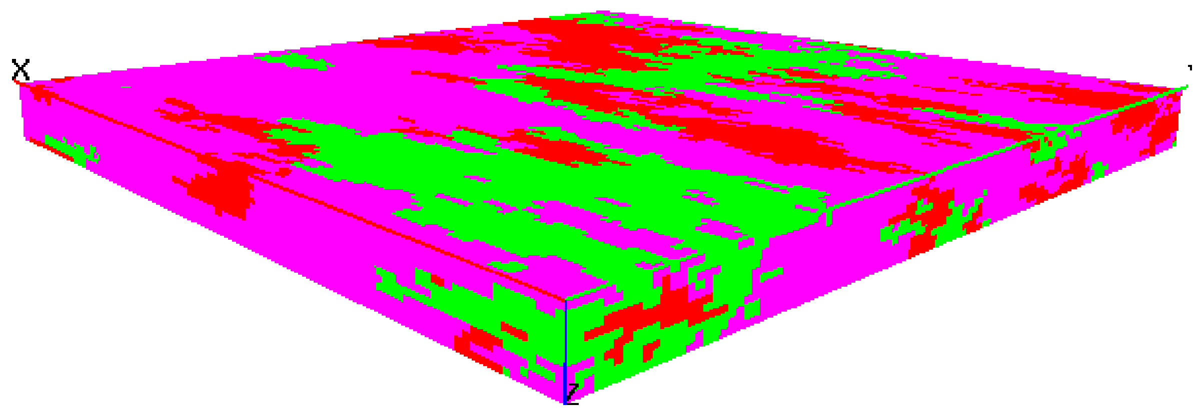
- Generate a 3D Gaussian field: X(s)
- Assign type ”i” according to thresholds:
  - $t_i(s) < X(s) < t_{i+1}(s) \Rightarrow$ type “
- Strict ordering





Indicator kriging

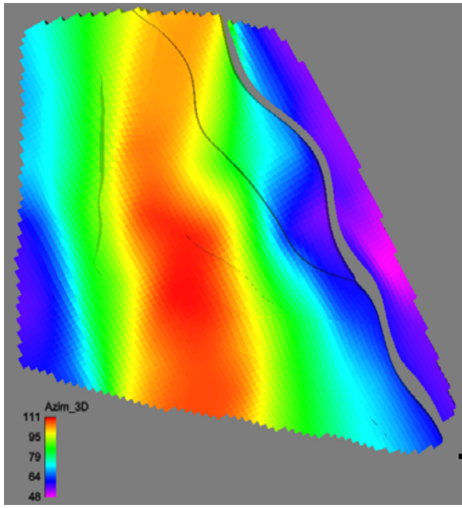
- Tries to calculate a probability for a type
- Uses kriging to interpolate probabilities



- Sequential simulation algorithm

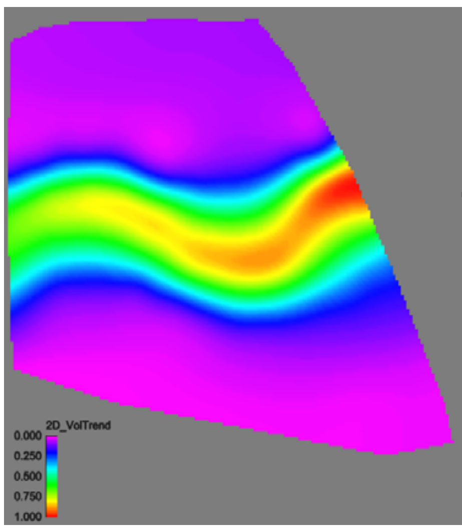
Indicator kriging

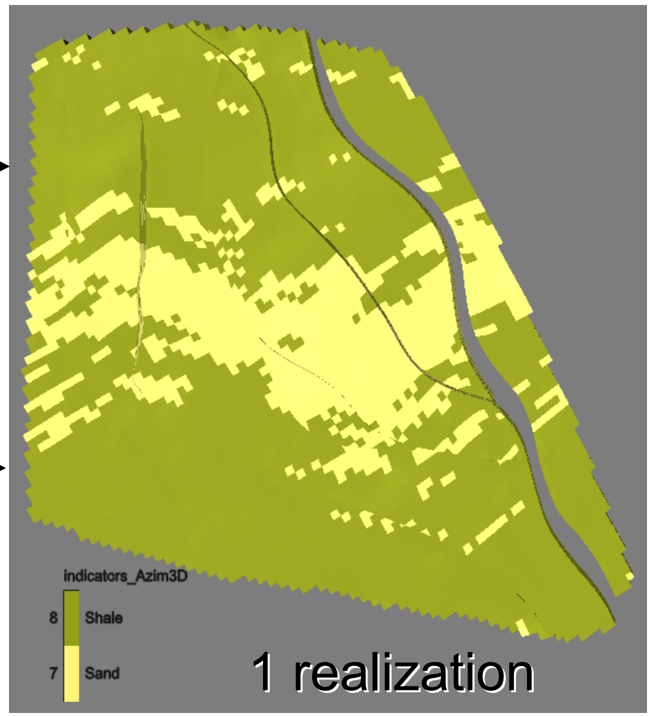
3D azimut trend



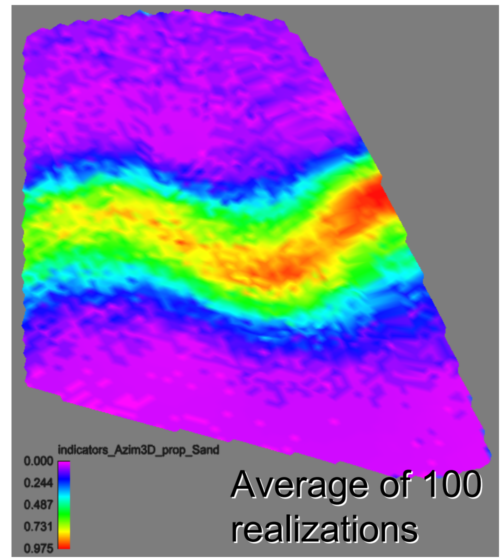
- ► In use on fields with 10 000 - 17 000 wells in Russia
- ► Robust volume fraction steering
- ► 1D/2D/3D or combined volume trends
- ► 3D trends on azimuth and variogram ranges
- ► Maintains continuous sand-layers or barriers if desired

3D volume trend





Indicators parameter

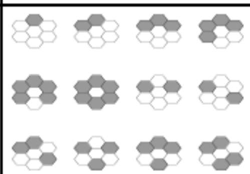


Sand fraction map

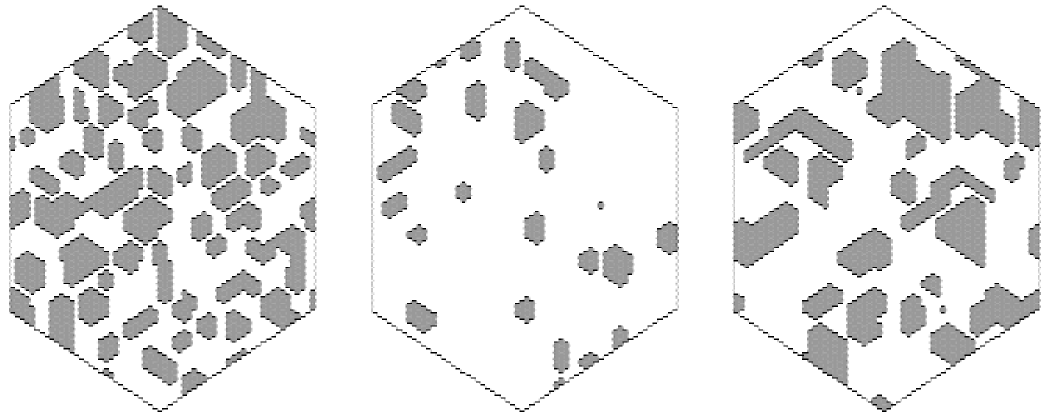
Markov random field

- Rich but abstract pixel based method
- MCMC algorithm for simulation
- Major problems:
  - Speed – MCMC is to slow
  - Hard to determine model
    - Estimation (only ML will work)
    - Abstract model makes it hard to specify manually
  - Phase transition makes it unstable
- Advantage: Consistent probabilistic model
  (Why is that an advantage?)

MRF specifications

conf. type	$V_C(z_C)$	configurations
foreground	θ_1	
concave	θ_2	
line	θ_3	
convex	θ_4	
sharp convex	θ_5	
other	θ_6	
background	θ_7	
edge backgr.	0	

- Realisations from second order neighbourhood model



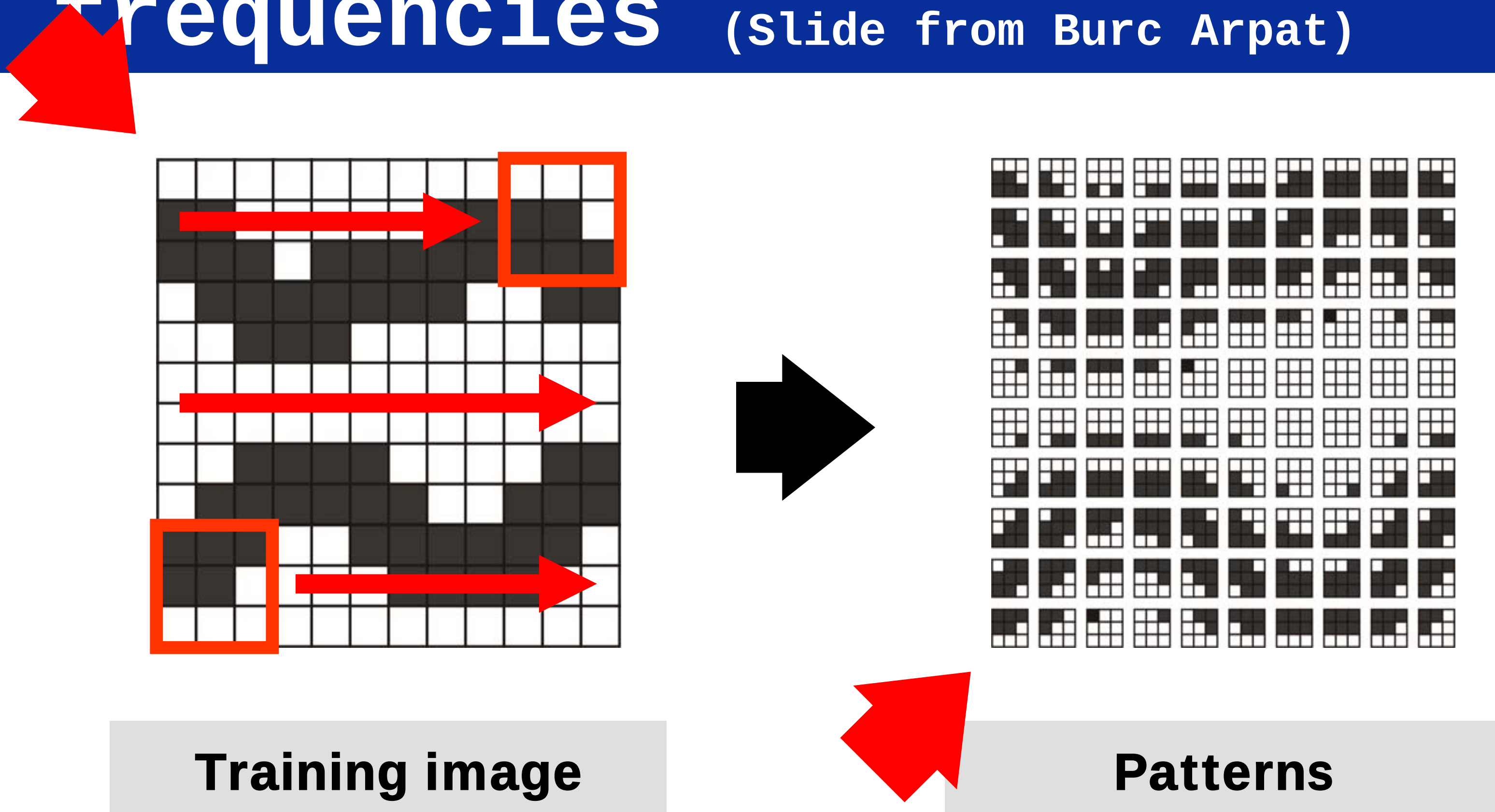
$$f(z) = c \cdot \exp\left\{ -\sum_{C \in \mathcal{C}} V_C(z_C) \right\}$$

Multipoint algorithms

- The Snesim algorithm (Stanford: Srivastava, Strebelle, Caers,…)
- Main idea is to:
  1. Capture geometric features in a training image:
     - Count pattern frequencies
  2. Sequential simulation:
     - Probabilities according to pattern frequencies

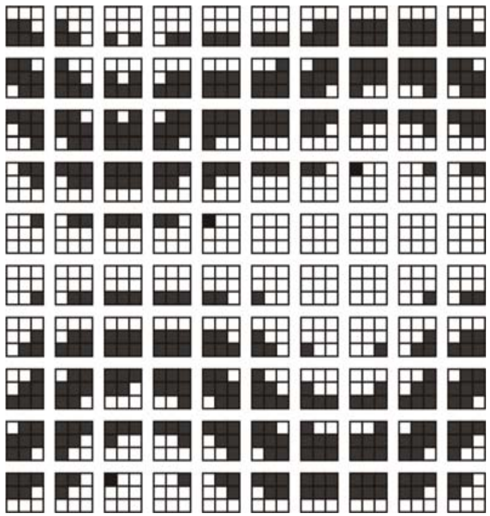
- Comparison to MRF:
  1. Estimate parameters in potentials
     - MLE
  2. Iterative MCMC simulation:
     - Conditional probabilities according to estimated model

Counting pattern frequencies (Slide from Burc Arpat)

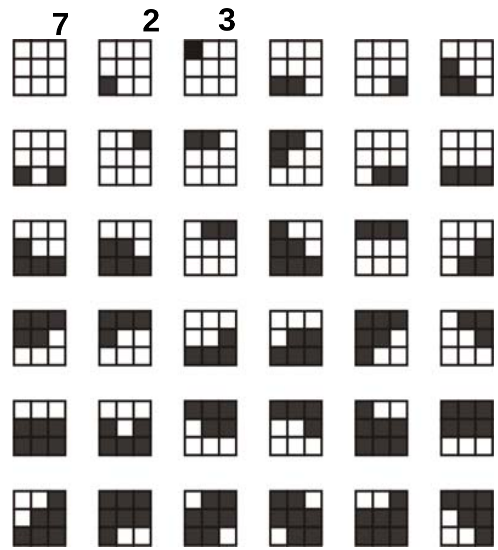


Step 1: Scan the training image using a template (window) to find all available geological patterns

Counting pattern frequencies (Slide from Burc Arpat)



Patterns



Pattern database

Step 2 : Process the patterns obtained from the training image to construct the pattern database

Note: Only 36(?) patterns out of $2^9 = 512$ possible patterns.

Only 100 possible patterns in 12×12 training image.

Sequential simulation

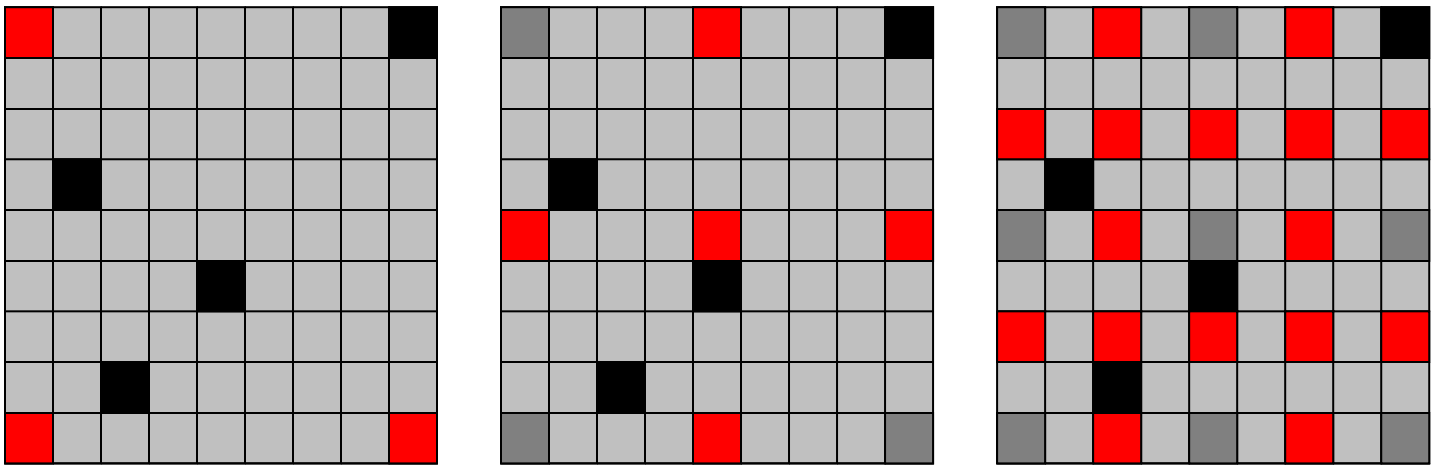
- Exact if

  $P(x_1,\ldots,x_n) = P(x_1)\cdot P(x_2|x_1)\cdot P(x_3|x_1,x_2)\cdot P(x_4|x_1,x_2,x_3)\cdots P(x_n|x_1,\ldots x_{n-1})$

- Necessary approximation: Only consider x's in a (small) neighborhood:

  $$P(x_k|x_1,\ldots x_{k-1}) \approx P(x_k|\partial(x_k))$$

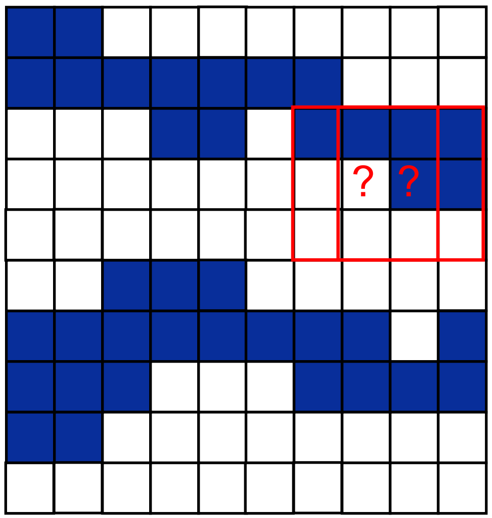
- Random path through grid follows a refinement scheme:



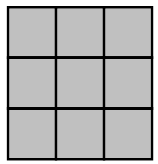
- Ensures good large-scale behavior

Simulation

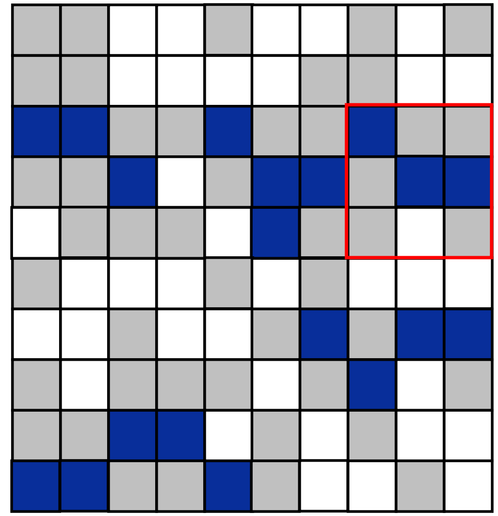
Training image



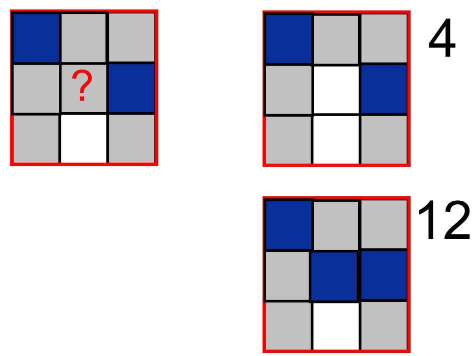
Template



Unfinished simulation



Patterns found in TI



Is there anything wrong with these frequencies/probabilities?

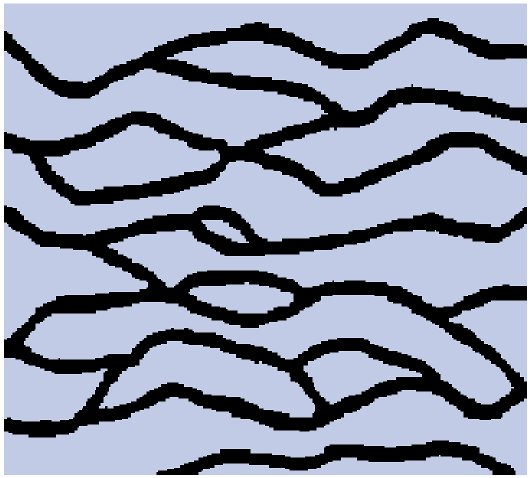
- Looks intuitively very nice
- Recall

$$P(x_1,\ldots,x_n) = P(x_1)\cdot P(x_2|x_1)\cdot P(x_3|x_1,x_2)\cdot P(x_4|x_1,x_2,x_3)\cdots P(x_n|x_1,\ldots,x_{n-1})$$

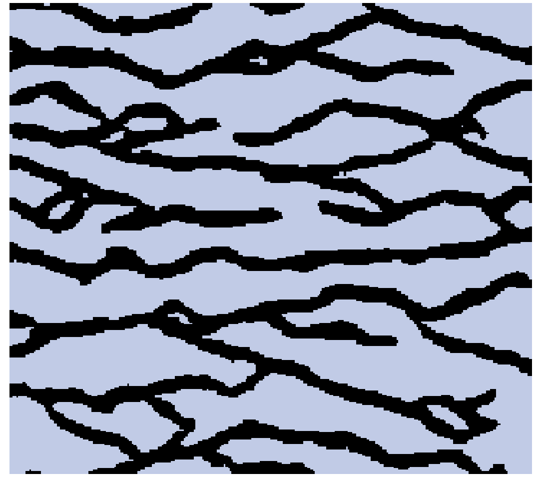
- The P's are estimated from training image
- …but we don't know $P(x_k|x_1,\ldots,x_{k-1})$
- We would need to marginalize:
  $P(x_k|x_1,\ldots x_{k-1}) = \Sigma_{x_{k+1}\in I}\cdots\Sigma_{x_n\in I} P(x_k|x_1,\ldots,x_{k-1},x_{k+1},\ldots,x_n)$
- We are unable to do that

SNESIM artefacts

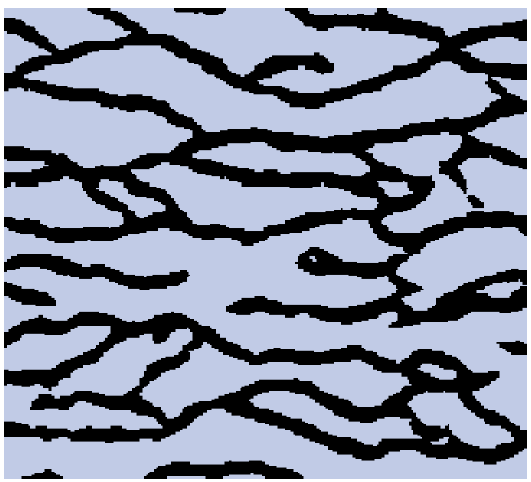
Training image



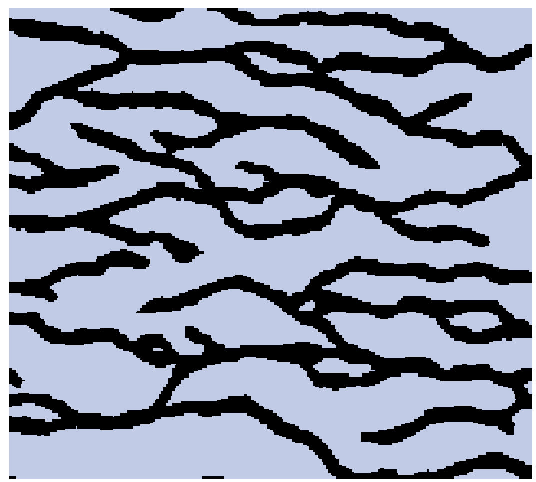
Realization 1



Realization 2



Realization 3

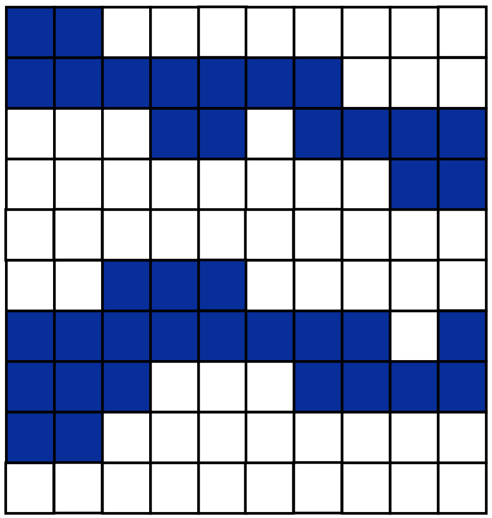


What goes wrong?

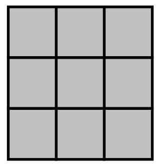
- Sequential methods encounter impossible situations since
  - Algorithm can’t detect future inconsistencies.
- Solution:
  - Node dropping: Conditioning data from earlier simulations are dropped.

Node dropping

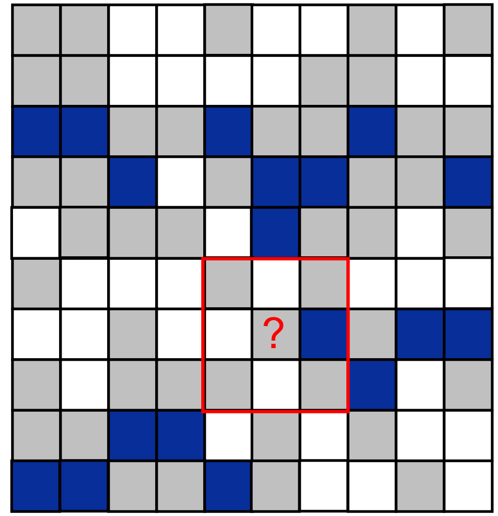
Training image



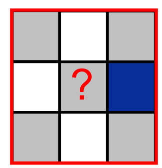
Template



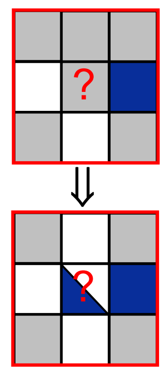
Unfinished simulation



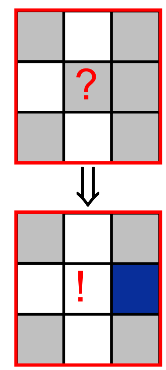
No pattern found in TI



Dropping white node



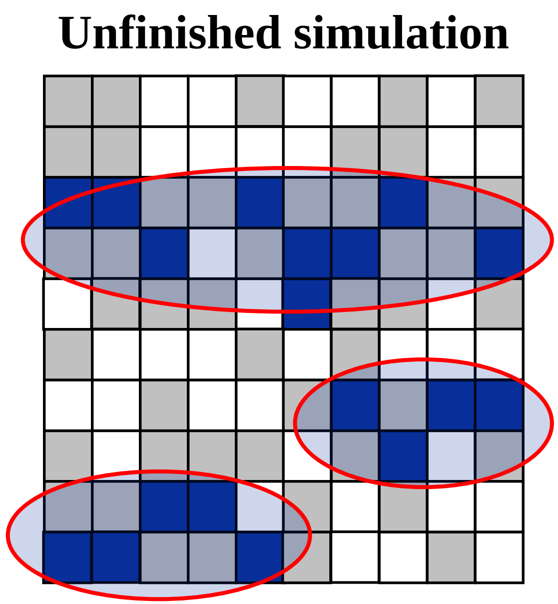
Dropping blue node



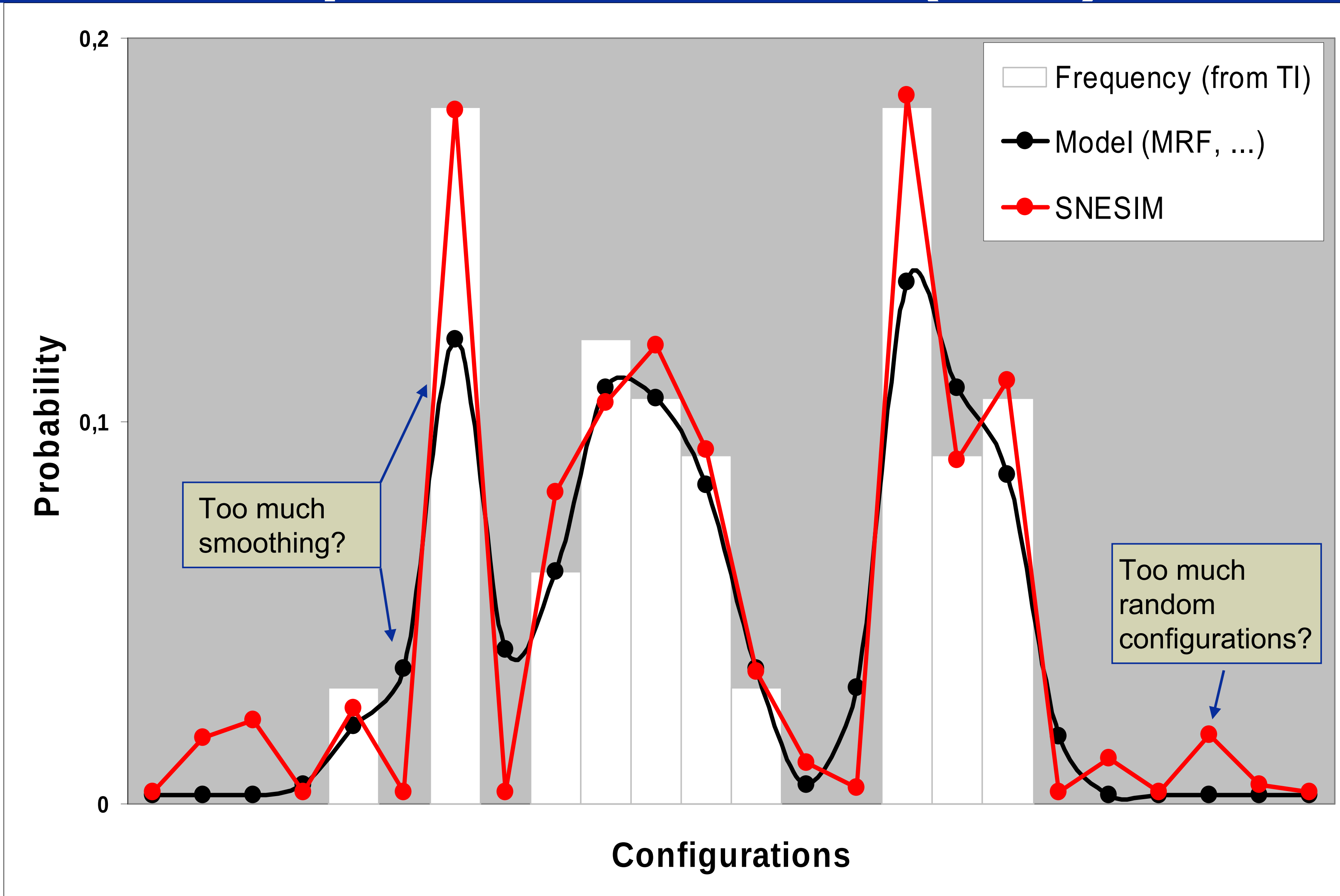
Arbitrary choice determines colour.

The reason for the conflict

- Three unfinished channels has started to form.
- Two are blocked by white areas.



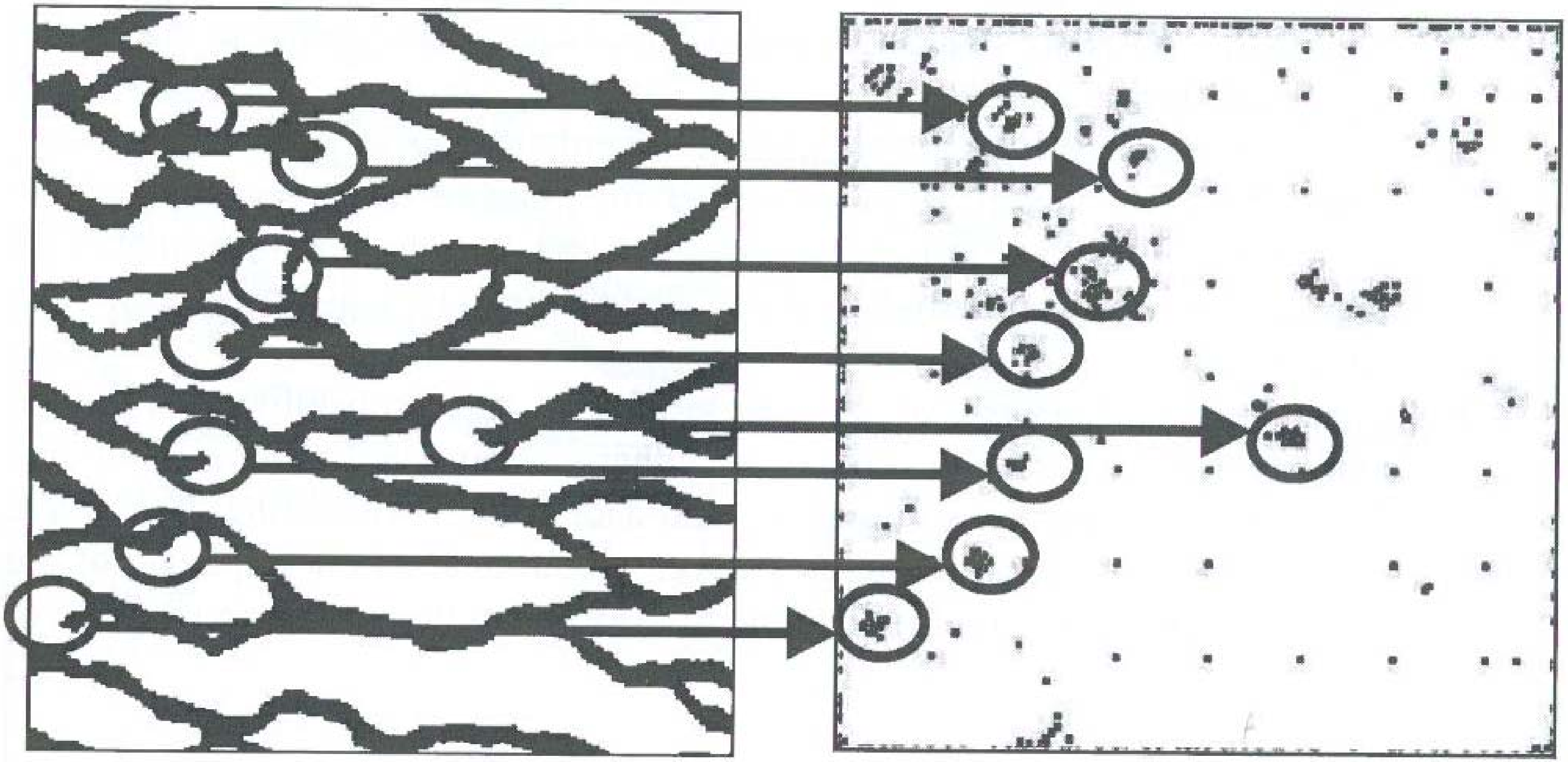
Conceptual illustration (1D!!)



Dead end areas have a lot of node dropping

Realization

Areas with less than 10 conditioning points



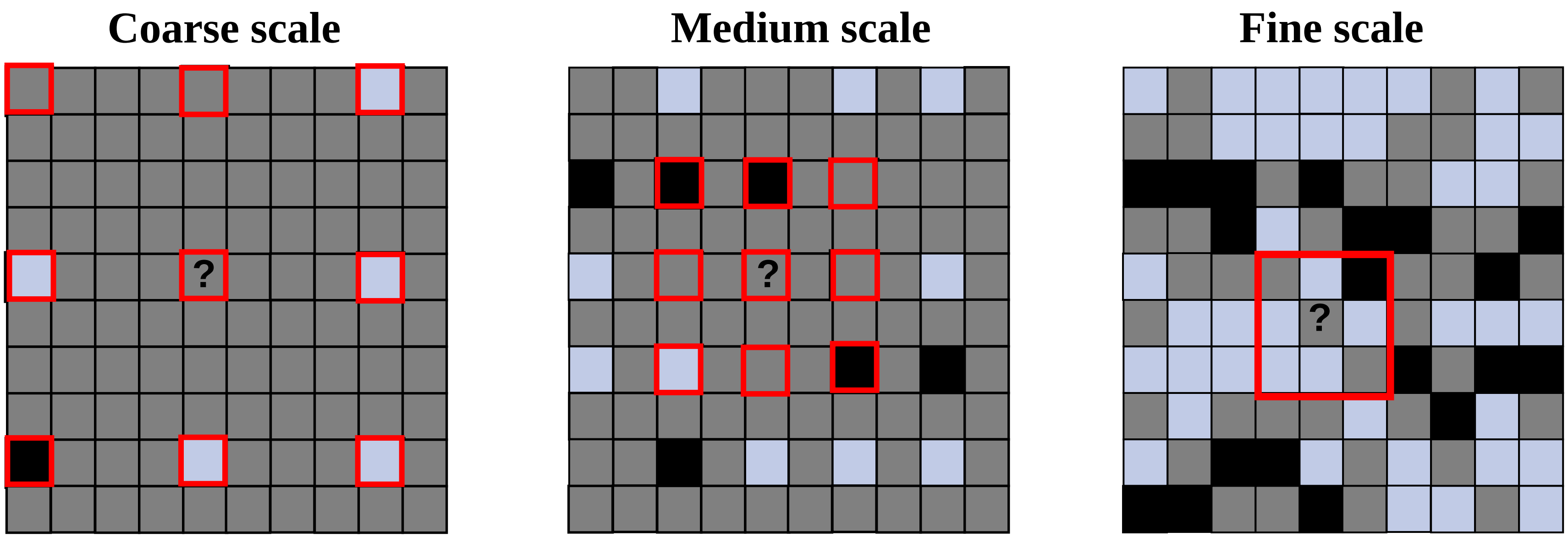
S. Strebelle and N. Remy, Geostatisitcs Banff 2004

Possible solution

- Delete previously simulated data that doesn't fit TI.
  - Only delete if a serious misfit to TI patterns occur.
- Deletion implies some iteration – previously simulated values must be re-simulated.

Multiple grids

- Refer to Tran(2004)
- Simulate on different scales to capture large scale features and do fine scale smoothing

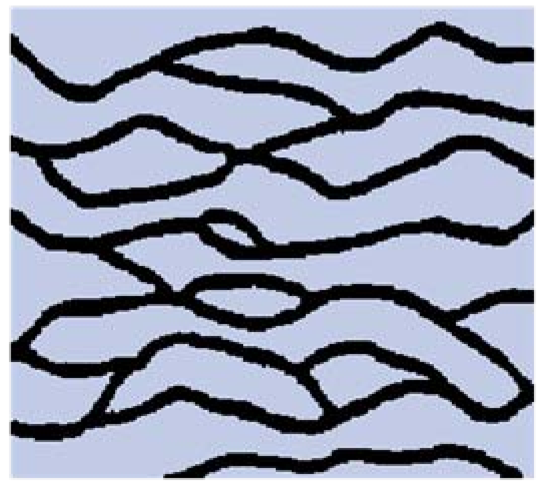


Example 1: Fluvial channels

Template

			49	45	50			
	57	37	29	25	30	39	59	
	38	22	13	9	14	23	42	
53	31	15	5	4	6	20	33	56
47	28	11	1	?	2	12	26	48
54	36	18	8	3	7	16	35	55
	44	21	17	10	19	24	40	
	60	43	34	27	32	41	58	
			51	46	52			

Training image



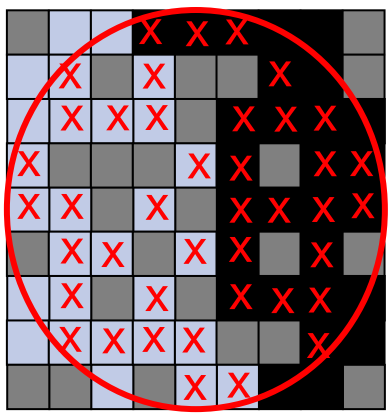
Grid size: 250 x 250

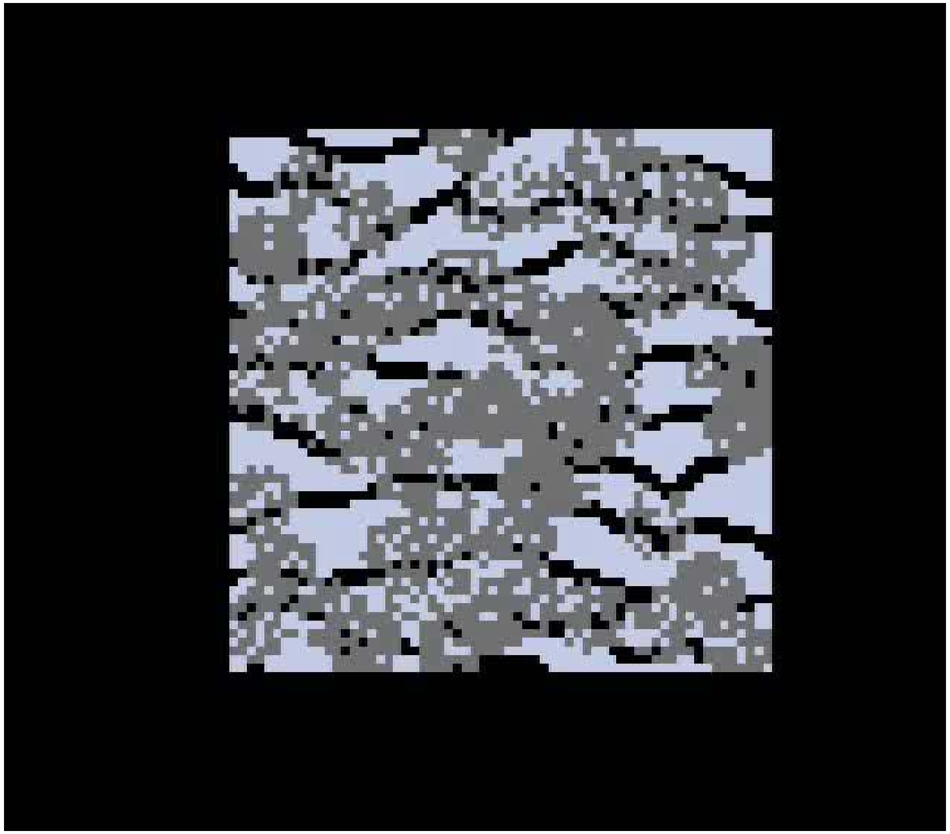
Number of grids = 3

Template size = 60

Delete all nodes in template

- If conflict, all sampled nodes in the template are deleted

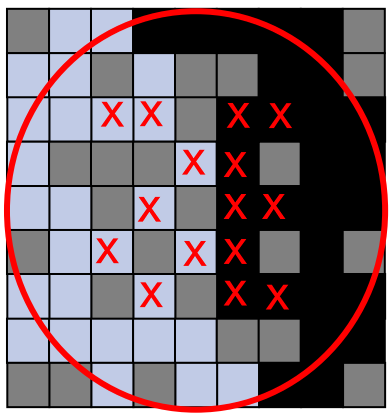




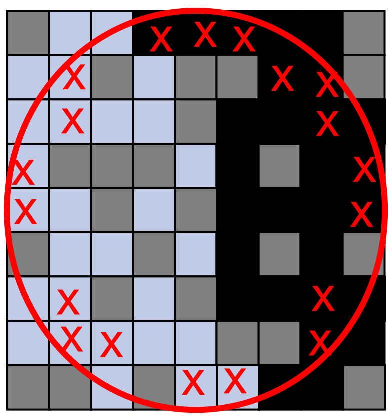
Delete nearest / most distant nodes

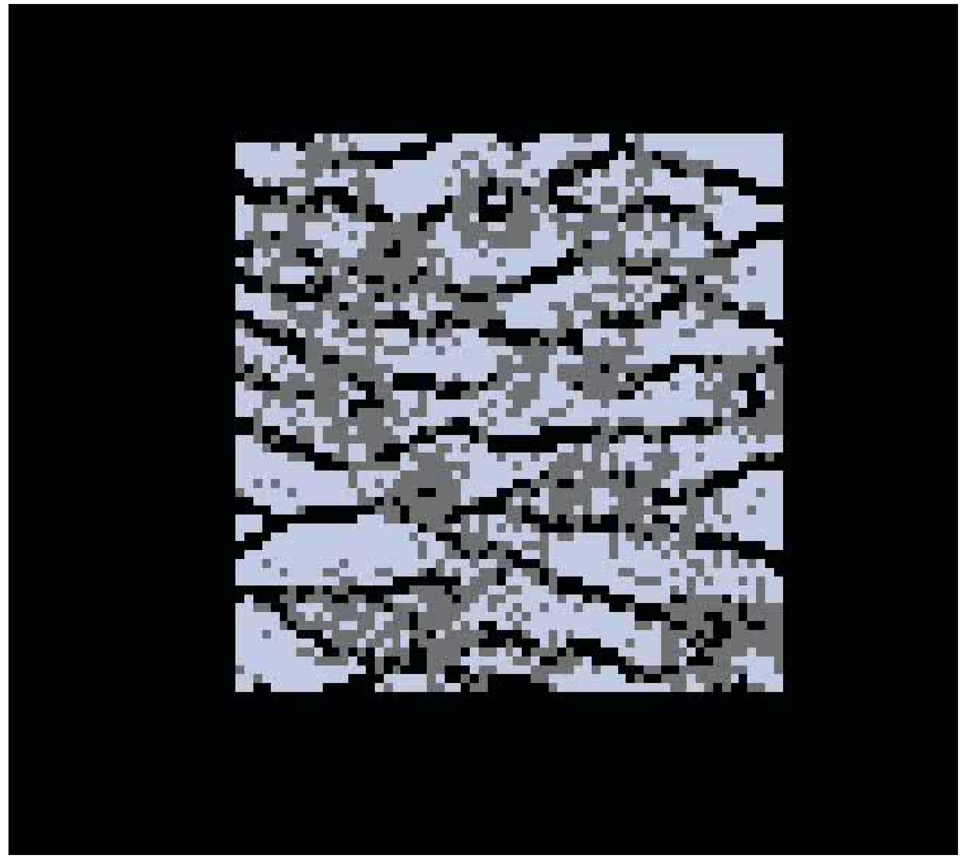
- Delete either nearest or most distant nodes

Nearest



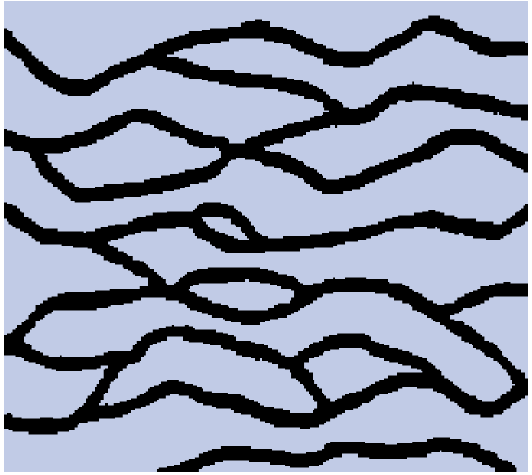
Most distant



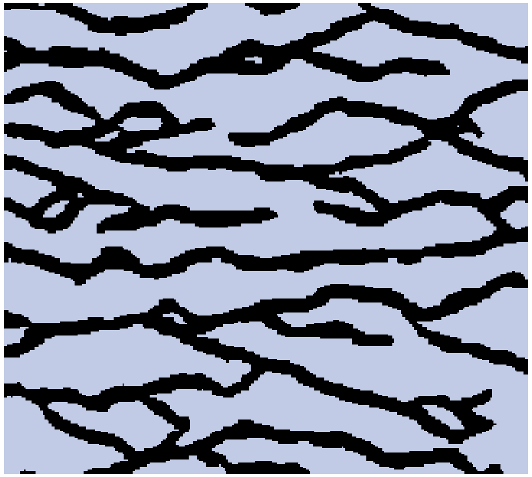


Visual comparison

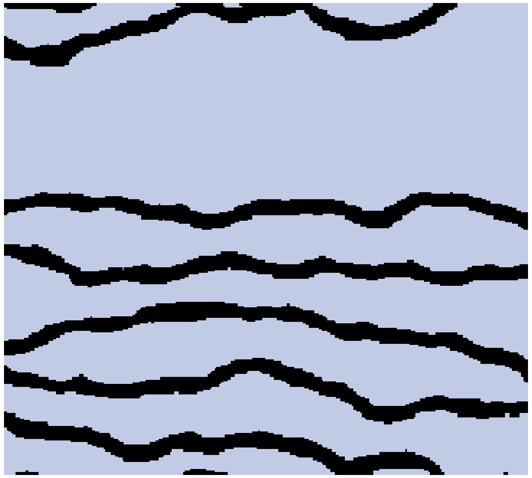
Training image



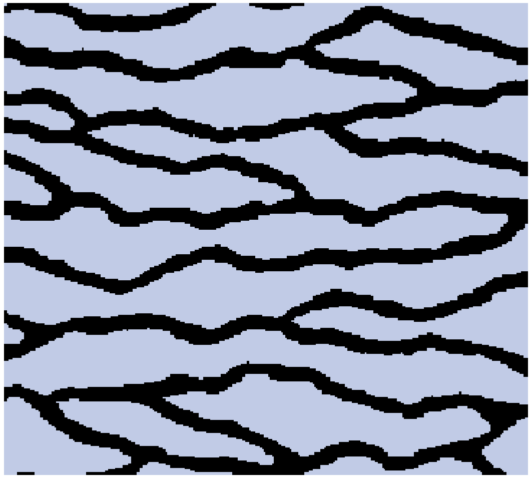
Node dropping



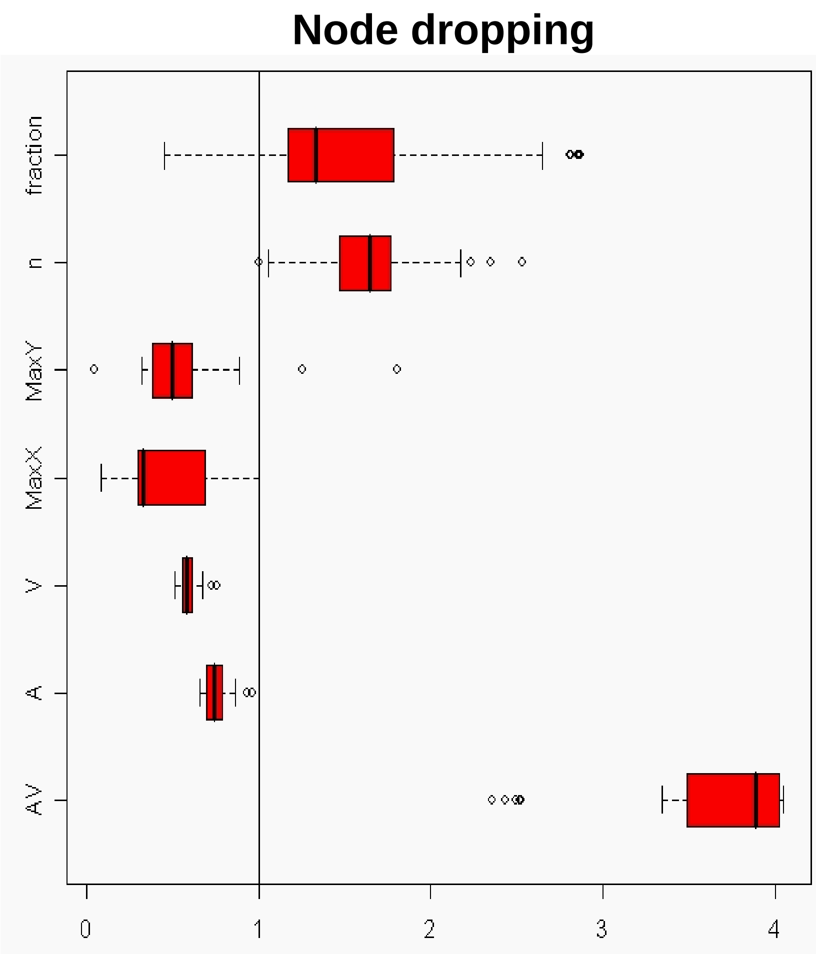
Delete all

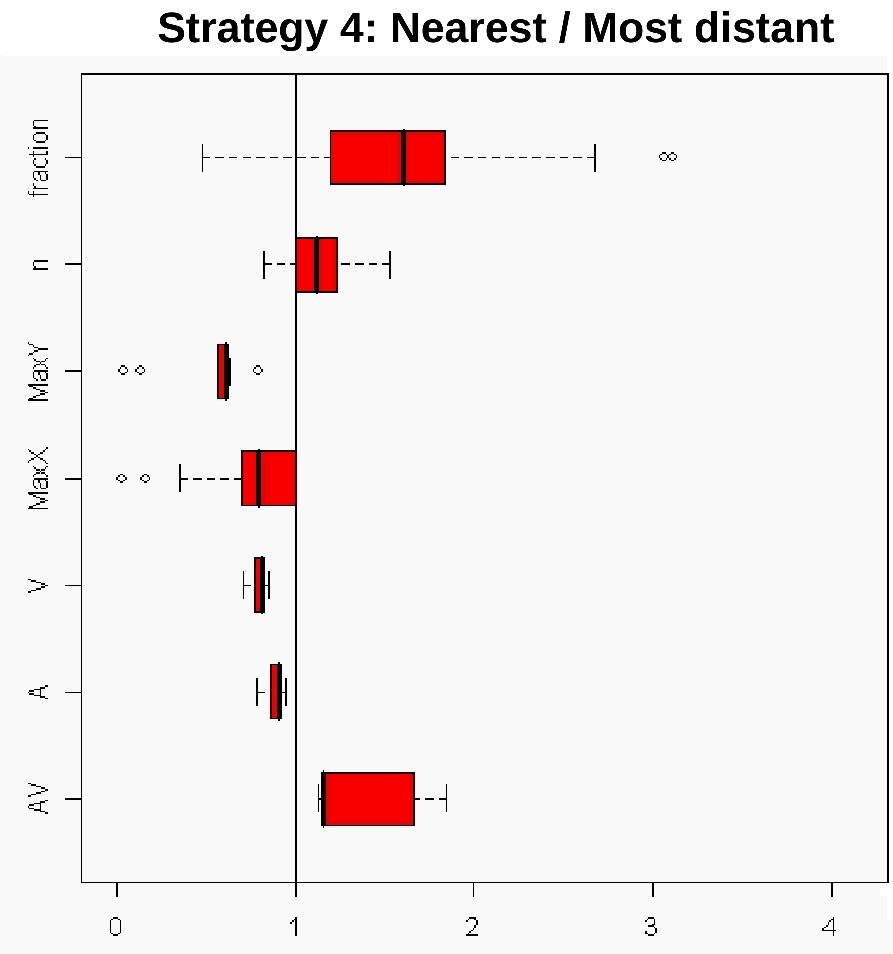


Delete near/far



Statistical analysis SNESIM and modified SNESIM





Simulation in practice

- Large variety
  - 5 realizations – 5000 realizations
  - The more the better ☺
- Approximations
  - Nothing is perfect – but it can still be very useful
- Consider the objectives
  - Stupid way of calculating π
  - Use it when easy, efficient or the only way
- Used for complex problems
  - High dimension
  - Complicated and important dependencies
  - Nested dependencies
  - Non-linearity