

Perfect simulation of Matérn Type III point processes

Mark Huber

Departments of Mathematics and Statistical Science
Duke University

17 Mar, 2009

Joint work with

- Robert Wolpert (Duke University)
- Jesper Møller (Aalborg University)

Some repulsive things

Spanish towns

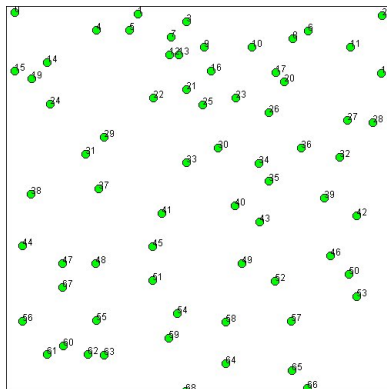


Pine trees

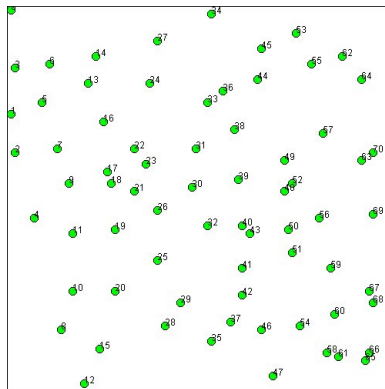


Plotting locations

Locations: Spanish towns

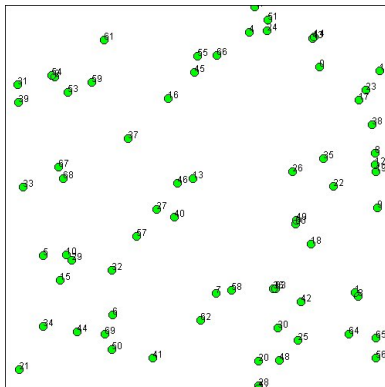


Locations: Swedish pines



Poisson process

Constant intensity: locations uniform



1 Modeling repulsive point processes

- Densities
- Matérn processes

2 Approximation for Matérn Type III

- Building a Markov chain
- Perfect simulation
- The product estimator

3 Extensions

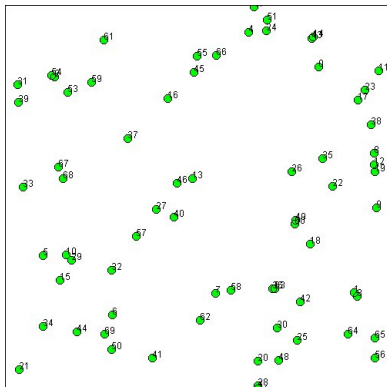
- Soft core
- Dealing with edge effects

Ways to model repulsion

Two different approaches:

- Create density with respect to Poisson point process
- Create algorithm that modifies Poisson point process

Poisson point process



Space S

Intensity measure $\lambda \cdot \mu(\cdot)$

For $A \subseteq S$, $\mathbb{E}[A] = \lambda \cdot \mu(A)$

What is a Poisson point process?

Divide region into tiny squares

- Probability square contains point...
- ...equals size of square times λ

To generate process:

- Draw $N \leftarrow \text{Poisson}(\lambda \cdot \mu(S))$
- Randomly place N points on S using $\mu(S)$
- (When μ is Lebesgue measure, points uniform over S)

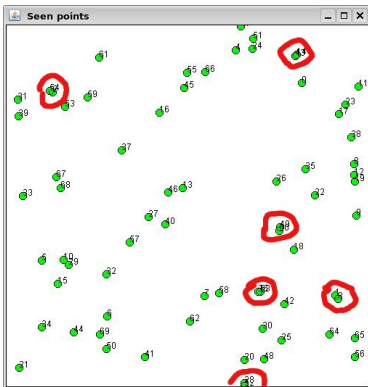
Example: density approach

Strauss process¹

γ := repulsion parameter in $(0, 1)$

R := radius of interaction

$$f(x) = \gamma^{\#\{(i,j): \text{dist}(x_i, x_j) < R\}} / Z$$



$$R = .02$$

$$f(x) = \gamma^6 / Z$$

¹Strauss 1975

Advantages

- Easy to write down
- Perfect simulation algorithms exist
- Can deal with edge effects
- Used for maximum likelihood or as prior for Bayesian inference

Disadvantages

- Unknown normalizing constant Z
- Has phase transition
- Makes Markov chain slow for big λ

More algorithmic²

- Start with Poisson point process
- Apply procedure to induce repulsive effect

Advantages

- By definition easy to simulate
- Known normalizing constant

Disadvantages

- Unknown density
- No density = no MLE = no posterior

²Matérn 1960

Generate Type I Matérn process

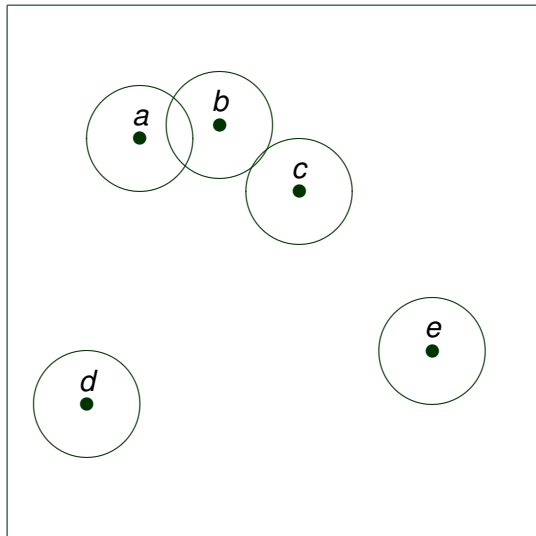
Input: $R, \lambda(\cdot)$ *Output:* x

- 1) **Draw** $A \leftarrow \text{Poisson point process}(\lambda(\cdot))$
- 2) **For** all $\{p, p'\} \subset A$ **do**
- 3) **If** $\text{dist}(p, p') \leq R$
- 4) **Let** $A \leftarrow A \setminus \{p, p'\}$

Thinned Poisson point process:

- Start with PPP
- Remove any pair of points within distance R of each other

Type I: Picture

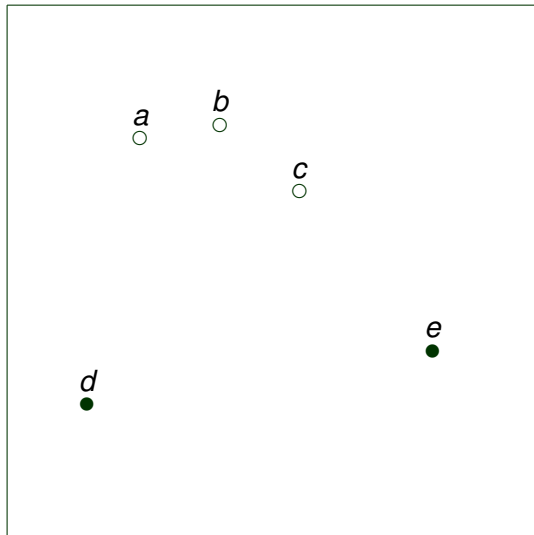


Circles of radius $R/2$

Circles touch =
points eliminated

Points a , b , c eliminated

Type I: After thinning



Call a, b, c *ghost points*

Call d, e *seen points*

Ghost points exert
invisible pressure

Problems

- Too many eliminations
- As $\lambda \rightarrow \infty$, # of points $\rightarrow 0$
- Need method that preserves some points

Generate Type II Matérn process

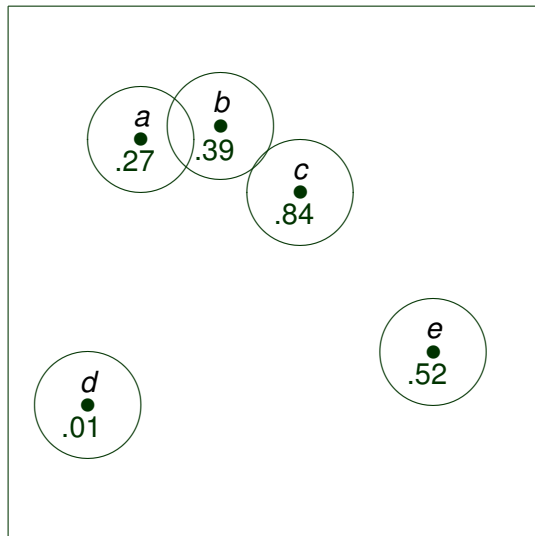
Input: $R, \lambda(\cdot)$ *Output:* x

- 1) **Draw** $A \leftarrow \text{Poisson point process}(\lambda(\cdot))$
- 2) **For** all points $p \in A$
- 3) **Draw** a time stamp $t_p \leftarrow \text{Unif}([0, 1])$
- 4) **For** all $\{p, p'\} \subset A$ do
- 5) **If** $\text{dist}(p, p') \leq R$ and $t_p < t_{p'}$
- 6) **Let** $A \leftarrow A \setminus \{t_{p'}\}$

Thinned Poisson point process:

- Start with PPP
- Remove point if within distance R of point born earlier

Type II: Picture

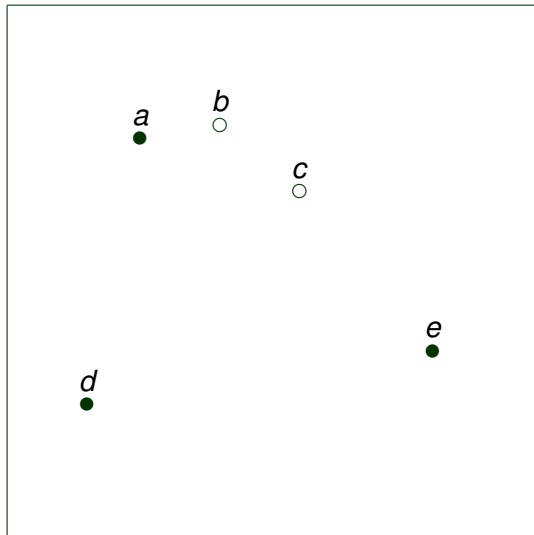


Circles of radius $R/2$

Point a eliminates b

Point b eliminates c

Type II: After thinning



Call b, c *ghost points*

Call a, d, e *seen points*

Ghost points exert
invisible pressure

Advantages

- First points survive
- Higher number of points than Type I

Problems

- Still can't write a density down
- Would like higher density of points

Generate Type III Matérn process

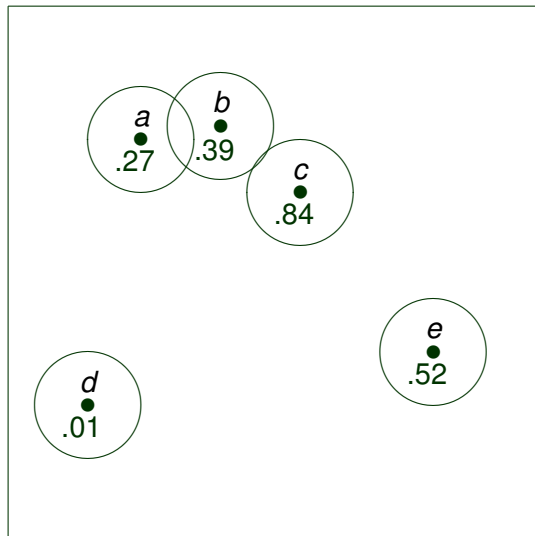
Input: $R, \lambda(\cdot)$ *Output:* x

- 1) **Draw** $A \leftarrow \text{Poisson point process}(\lambda(\cdot))$
- 2) **For** all points $p \in A$
- 3) **Draw** a time stamp $t_p \leftarrow \text{Unif}([0, 1])$
- 4) **Let** $L \leftarrow \{p_1, p_2, \dots, p_{\#A}\}$ **where** $t_{p_1} \leq t_{p_2} \leq \dots t_{p_{\#A}}$
- 5) **While** $L \neq \emptyset$
- 6) **Let** p be entry of L with smallest time stamp
- 7) **Let** $A \leftarrow A \setminus \{q : t_p < t_q\}$

Thinned Poisson point process:

- Start with PPP
- Remove point if within distance R of point born earlier that hasn't already been eliminated

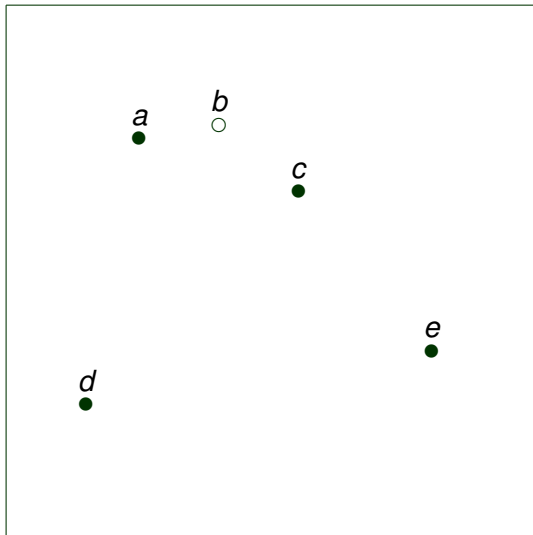
Type III: Picture



Circles of radius $R/2$

Point a eliminates b

Type III: After thinning



Call b *ghost point*

Call a, c, d, e *seen points*

Ghost points exert
invisible pressure

Type III: Comments

Advantages

- More points left to be seen
- More than twice density of Type II
- Can write a density
- Random Sequential Adsorption (RSA) model in Poisson limit

Needed pieces to use Type III:

- Write down the density
- Build Markov chain for density
- Build perfect sampler around Markov chain
- Build product estimator to utilize samples

1 Modeling repulsive point processes

- Densities
- Matérn processes

2 Approximation for Matérn Type III

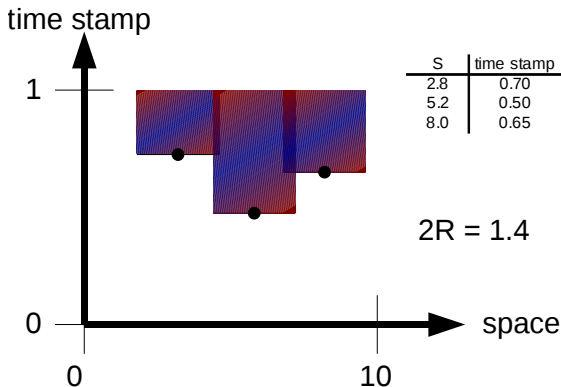
- Building a Markov chain
- Perfect simulation
- The product estimator

3 Extensions

- Soft core
- Dealing with edge effects

Casting shadows

The seen points cast shadows where the ghosts live:



The density

The density of point locations and time stamps (with respect to a Poisson point process with intensity $\mu(\cdot)$) is

$$f(x, t|\theta) = \exp(\mu(S)[1 - \lambda])\lambda^{\#x} \exp(\lambda \cdot \mu(\text{shadow})), \theta = (\lambda, R)$$

Remarks:

- $\exp(\mu(S)[1 - \lambda])$ resets intensity from $\mu(\cdot)$ to $\lambda\mu(\cdot)$
- Only the shadow is a function of t

$$g(x|\theta) = \exp(\mu(S)[1 - \lambda])\lambda^{\#x} \int_{t \in [0,1]^{\#x}} \exp(\lambda \cdot \mu(\text{shadow})).$$

Usual situation:



$$g(x) = \frac{w(x)}{Z}, \quad Z = \int_{\Omega} w(x) \, dx$$

- Typically Z difficult to compute (# P complete)
- With Matérn: know normalizing constant, integration in weight!

Our Monte Carlo approach

- Generate t to go along with x
- Use product estimator to approximate $g(x|\theta)$

How to draw time stamps given locations?

Markov chain Monte Carlo

- 1 Construct a Markov chain with stationary distribution matching target distribution
- 2 Under mild conditions (ϕ -irreducibility, aperiodicity) can guarantee limiting distribution matches stationary distribution
- 3 Run chain “for a long time” from arbitrary initial state to obtain samples

Once have density, can use Metropolis method

- Metropolis³ is a protocol for building Markov chains with target stationary distribution
- At state t , propose move (uniformly) to new state t'
- Accept move with probability:

$$\min \left\{ 1, r(t', t) := \frac{f(x, t' | \theta)}{f(x, t | \theta)} \right\}$$

- Tricky part:

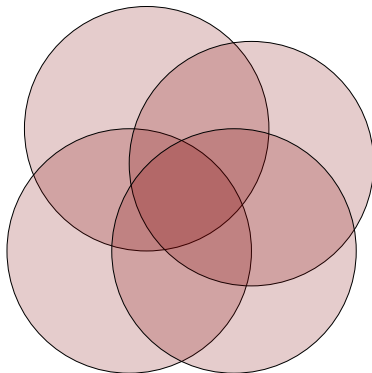
$$r(t, t') = \exp(\lambda[\mu(\text{shadow under } t') - \mu(\text{shadow under } t)])$$

³Metropolis, Rosenbluth, Rosenbluth, Teller & Teller 1953

Why tricky?

Keep it simple: only change t_p for single point p

μ (change in shadow) involves (even in 2D) intersections of circles:

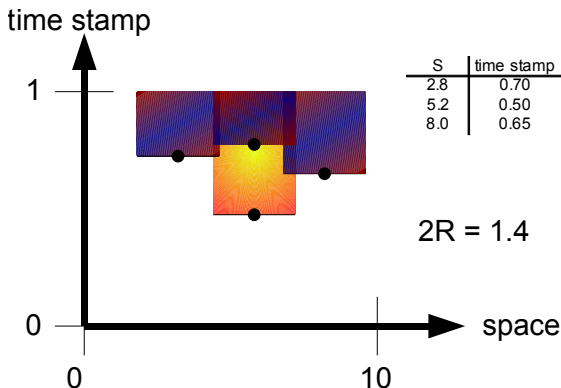


Avoiding the ratio

Easy case: $t'_p \leq t_p$ (so point born earlier) means $r(t, t') \geq 1$

Hard case: $t'_p > t_p$, which decreases shadow

$$r(t, t') = \exp(-\lambda(\mu(\text{orange region})))$$



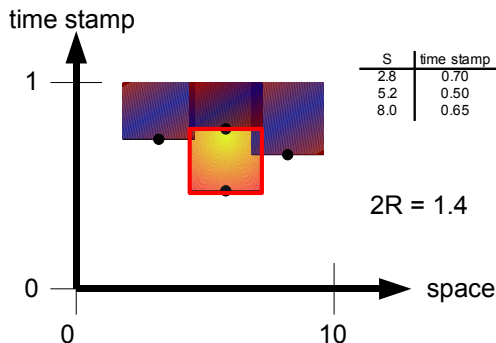
A Poisson subprocess

Idea #1:

$$\exp(-\lambda(\mu(\text{orange region}))) = \mathbb{P}(\text{PPP in orange region has 0 points})$$

Idea #2:

Draw PPP in orange region by thinning PPP in larger cylinder



How to take a step in Markov chain

Metropolis step

Input: seen points x , current time stamps t

Output: next time stamps t

- 1) **Draw** $i \leftarrow \text{Unif}(\{1, \dots, \#(x)\})$
- 2) **Draw** $t'_i \leftarrow \text{Unif}([0, 1])$
- 3) **If** $t'_i \leq t_i$
- 4) **Let** $t_i \leftarrow t'_i$
- 5) **Else**
- 6) **Draw** $W \leftarrow \text{Poisson point process over } [t'_i, t_i] \times B_r(x_i)$
- 7) **If** $\#\{W \cap \text{change in shadow}\} = 0$ **then**
- 8) **Let** $t_i \leftarrow t'_i$

Goal

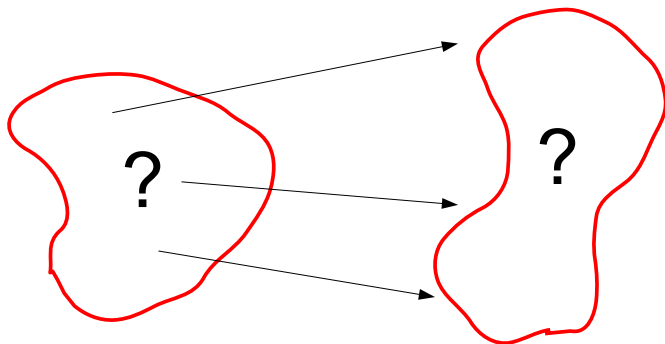
- Draw samples exactly from stationary distribution
- No need to know mixing time of chain

Drawback

- Running time is random
- Example: Acceptance/Rejection

Bounding chains⁴

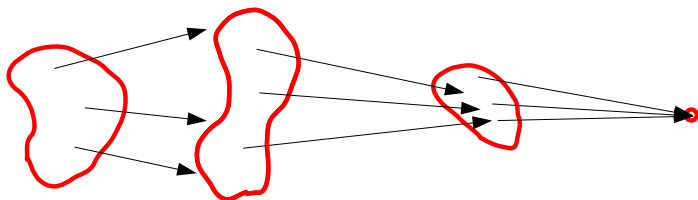
At each step look at possible states for next step:



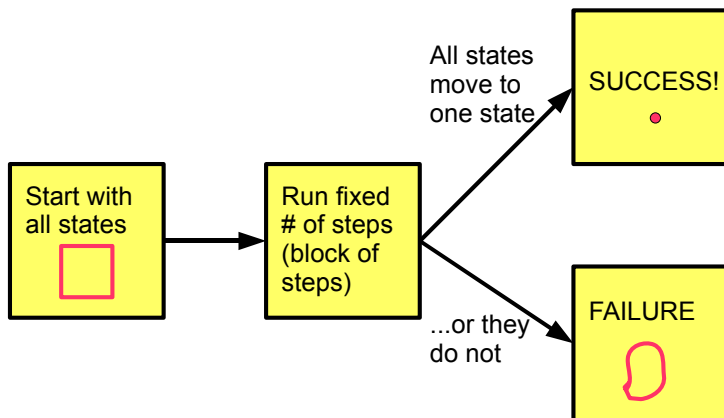
⁴Huber 1998, Häggström & Nelander 1998, Huber 2004

Bounding chains Part II

At each step look at possible states for next step:



Blocks either success or failure



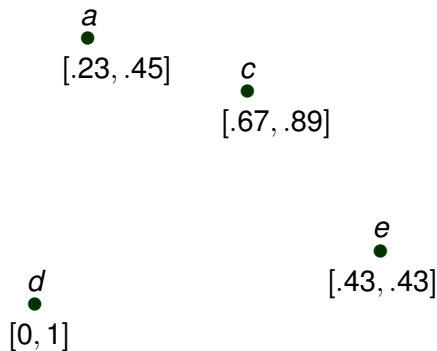
Coupling from the past⁵

- Say have method for deciding if block is success or failure...
- ...CFTP turns block method into method for generating exactly stationary draws
- Running time = time to generate block/probability block success
- No need to know mixing time of Markov chain
- Run success followed by geometric number of failure blocks

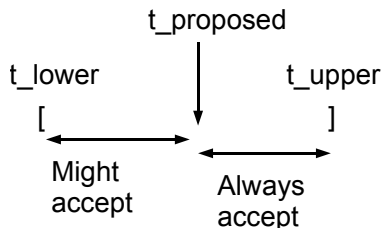
► Details of CFTP

⁵Propp & Wilson 1996

Using intervals to bound time stamps



Updating interval



For “might accept”

- Draw Poisson process over largest possible region based on other intervals
- If empty, accept move for all points below proposed t
- Gives proof that chain is *antimonotonic*⁶

⁶Kendall 1998, Kendall & Møller 2000, Häggström & Nelander 1997

Approximating the likelihood

What we have so far:

- A density for Matérn III that we want to approximate
- A Markov chain for generating approximate samples
- A perfect simulation algorithm for generating exact samples

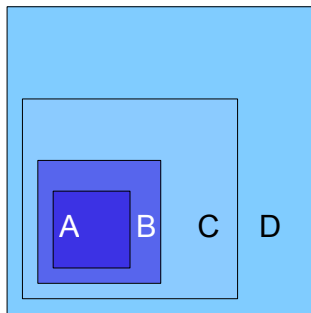
What we want:

- A method for turning samples into approximation of density
- Method: product estimator

Product estimator

A means for estimating sizes by looking at successive products:

$$\mu(D) = \mu(A) \frac{\mu(B)}{\mu(A)} \cdot \frac{\mu(C)}{\mu(B)} \cdot \frac{\mu(D)}{\mu(C)}$$



In context of Matérn III:

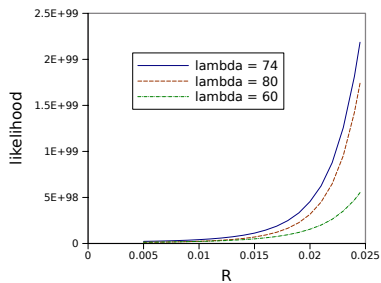
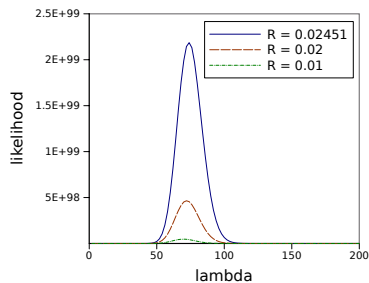
- Smallest set is $\lambda = 0$: $g(x|(0, R)) = \exp(\mu(S))$
- Create sequence $0 = \lambda_0 \leq \lambda_1 \leq \dots \leq \lambda_n = \lambda$
- Make $\lambda_i - \lambda_{i-1} = 1/\mu(S)$
- Then $g(x|(\lambda_i, R))/g(x|(\lambda_{i-1}, R)) \leq e$
- Need $O(n^2(1/\epsilon^2) \ln(1/\delta))$ samples for $1 + \epsilon$ -approximation with probability at least $1 - \delta$

$$\frac{g(x|(\lambda_i, R))}{g(x|(\lambda_{i-1}, R))} = \exp(\mu(S)(\lambda_{i-1} - \lambda_i)) C \exp((\lambda_i - \lambda_{i-1})\mu(\text{shadow}))$$

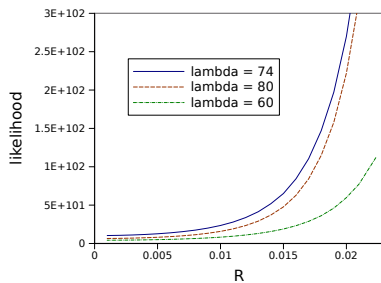
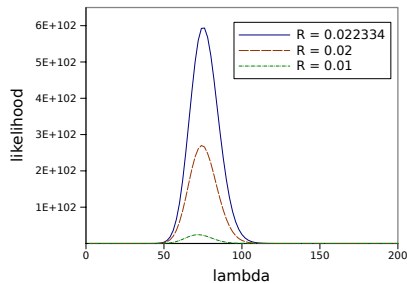
To estimate this last part without finding $\mu(\text{shadow})$:

- Use thinning trick from before
- Generate Poisson process with intensity $(\lambda_i - \lambda_{i-1})\mu(S)$ on S
- Only keep points that lie in shadow
- If no points remain, count as success
- $\mathbb{P}(\text{success}) = \exp((\lambda_i - \lambda_{i-1})\mu(S))$

Results: towns

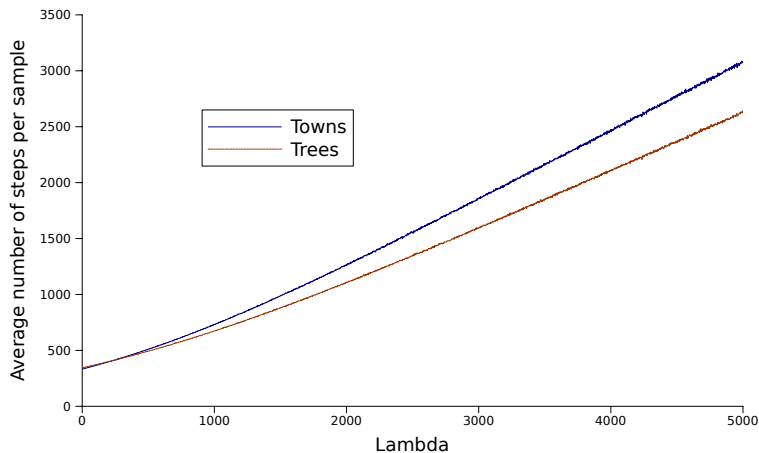


Results: trees



Running time per sample

Appears to be no phase transition



1 Modeling repulsive point processes

- Densities
- Matérn processes

2 Approximation for Matérn Type III

- Building a Markov chain
- Perfect simulation
- The product estimator

3 Extensions

- Soft core
- Dealing with edge effects

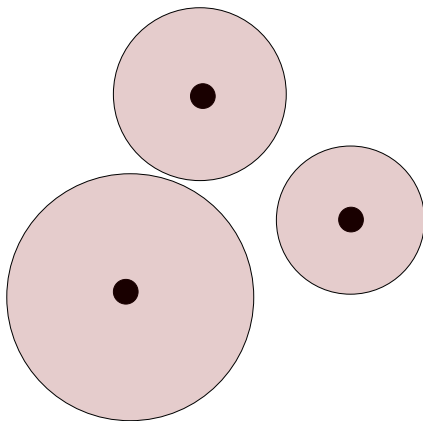
There are many ways to extend this work:

- Soft core—allowing some points to be closer than others
- Removing edge effects
- Discretizing

Goal: building model that matches actual distribution of distances

Several approaches

- Strauss model: penalize rather than forbid
- Individual values of R_i for point x_i



Dealing with edge effects

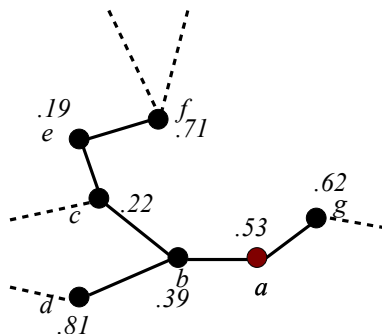
- Matérn's work all on finite space S
- Want method for \mathbb{R}^d dimensional analogue
- Look at finite window on infinite plane

Start with Poisson Point Process in a plane

- Connect any two points within distance R
- Track back from node until sure in or out of Matérn III

Finite window: example

e kills c so b survives, kills a

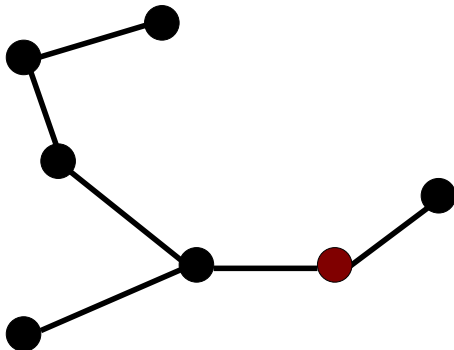


e is three steps away from a with $t_e \leq t_c \leq t_b \leq t_a$

Exponential versus Factorial

Consider graph of nodes within R

- Number of nodes k steps away grows exponentially
- Chance node k away matters is $1/k!$



Some data essentially discretized

- Tried running on neural spike train data
- Data only given to nearest millisecond
- Introduced artificial repulsion of 1 millisecond
- Need discrete version to apply to this type of data

Protocols for perfect simulation

- Here used coupling from the past⁷ and bounding chains⁸
- Randomness Recycler⁹
- Partial Recursive Acceptance/Rejection
- Sequential Acceptance/Rejection ($O(n^3)$ faster than Markov chains¹⁰)

Monte Carlo methods for permutations

- Nonmarkovian couplings
- Machine learning applications
- Nonparametric convex rank tests

⁷Propp & Wilson 1996

⁸Huber 2004

⁹Fill & Huber 2001

¹⁰Huber & Law 2008





For Matérn Type III processes

- This work built a density for the process
- Created a Metropolis Markov chain for the density
- Created a perfect simulation method for the chain to obtain exact samples with experimentally near linear (in intensity) run time
- Created a product estimator to use the samples to approximate the density

The result

- A method for likelihood estimating and posterior inference using the model of Matérn Type III processes
- Apparently no phase transition behavior

References

-  HUBER, M. (2004).
Perfect sampling using bounding chains.
Ann. Appl. Probab. **14**, 734–753.
-  HUBER, M. AND WOLPERT, R.L. (2008)
Likelihood-based inference for Matérn type III repulsive point processes
submitted
-  MATÉRN, B. (1986).
Spatial Variation, vol. 36 of *Lecture Notes in Statistics*.
New York, NY: Springer-Verlag, 2nd ed.
(first edition published 1960 by Statens Skogsforsningsinstitut,
Stockholm).
-  WILSON, D. B. (2000).
How to couple from the past using a read-once source of randomness.
Random Struct. Algor. **16**, 85–113.

Coupling from the past details

Let K^t be the kernel of t steps in Markov chain. For π stationary:

$$\pi K^t = \pi.$$

A block of t steps is either a SUCCESS block or FAILURE block

Let K^S be the kernel conditioned on SUCCESS

Let K^F be the kernel conditioned on FAILURE

Let p be the probability that a block is a SUCCESS

Then

$$K^t = pK^S + (1 - p)K^F.$$

From last slide:

$$\pi K^t = \pi, \quad K^t = pK^S + (1 - p)K^F.$$

Since SUCCESS block moves all states to same state,

$$\pi K^S = K^S,$$

in other words: SUCCESS blocks destroy memory of past. This gives:

$$\pi = \pi K^t = pK^S + (1 - p)\pi K^F.$$

Combining the facts from last two slides:

$$\begin{aligned}\pi &= \pi K^t \\ &= \pi(pK^S + (1-p)K^F) \\ &= pK^S + (1-p)\pi K^F \\ &= pK^S + (1-p)[pK^S + (1-p)\pi K^F]K^F \\ &= pK^S + (1-p)pK^S K^F + (1-p)^2\pi K^F K^F \\ &\vdots \\ &= pK^S + (1-p)pK^S K^F + (1-p)^2pK^S K^F K^F + \dots\end{aligned}$$

So to get draw from π , run a success block followed by G failure blocks, where $G \sim \text{Geo}(p)$.

Similar nature to acceptance/rejection

- CFTP related to acceptance/rejection (Fill's algorithm)
- Other modifications of A/R exist for perfect sampling
- All share 1) random run time, 2) generate exact samples

Read Once Coupling from the Past¹¹

Call success block S , failure block F

- Blocks independent of each other
- Result: Blocks look like Bernoulli trials $SSSF SFSFFFSS$
- Each state in Markov chain before an S block is stationary
- Why? Because it is an S block followed by geometric number of F blocks

► Go back

¹¹Wilson 2000