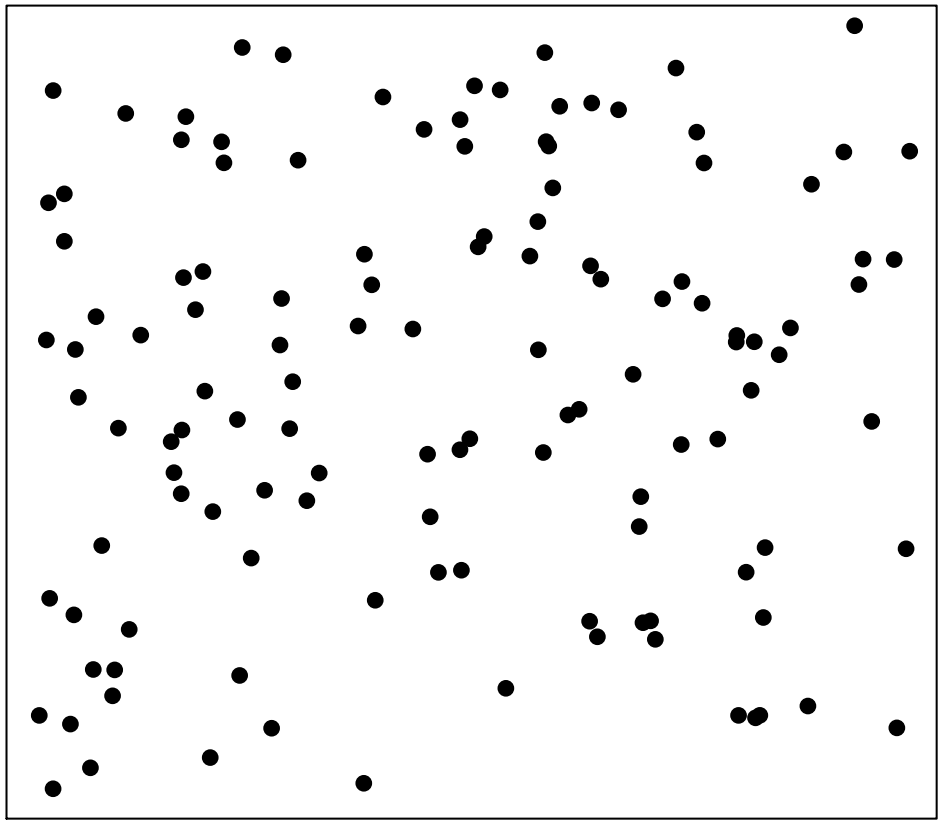
Sub-sampling using Determinantal Point Processes

Simon Barthelme, Pierre-Olivier Amblard, Nicolas Tremblay.
Gipsa-lab, CNRS

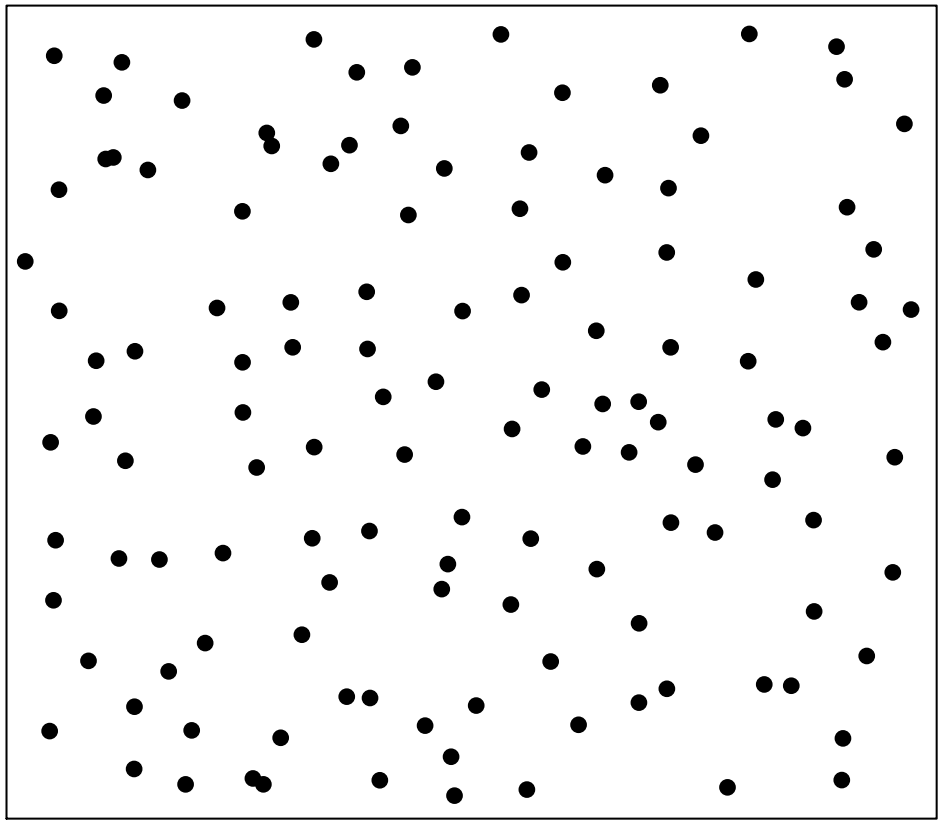
20th August 2018

Where DPPs come from



Non-interacting particles ("perfect gas", a.k.a. IID sampling)

Where DPPs come from



Fermions (Fermion point process, Macchi, 1976)

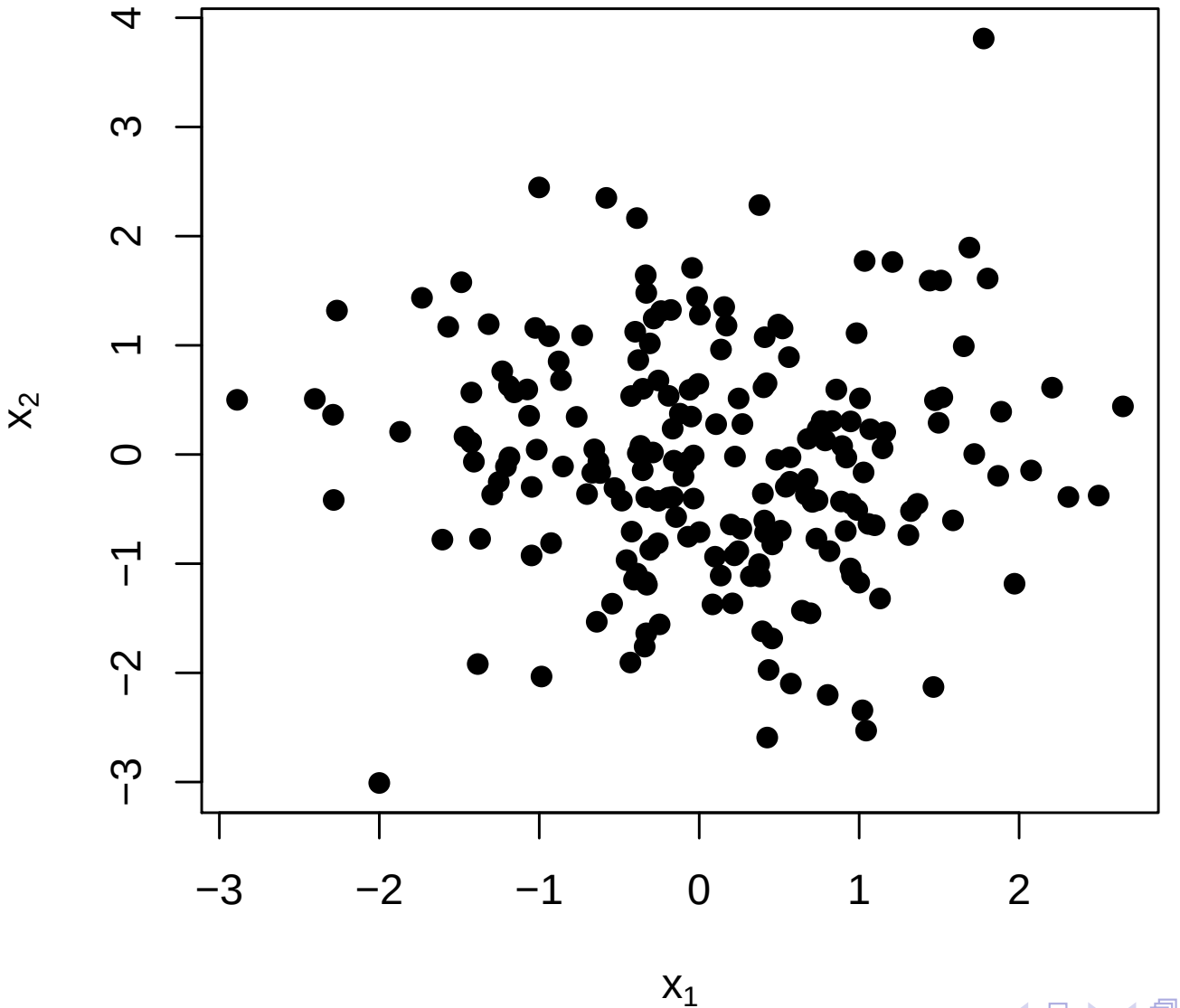
Variance reduction in importance sampling

- Many Monte Carlo methods try to achieve variance reduction by *enforcing diversity* in the samples
- Determinantal Point Processes provide a generic way of producing diverse subsets (Kulesza and Taskar, 2012)
- They have the advantage of tractability (compared to other point processes with repulsion).
- I will introduce DPPs briefly
- I will show applications to *graph sampling* and *coresets*
- Pointers to theoretical work at the end + challenges

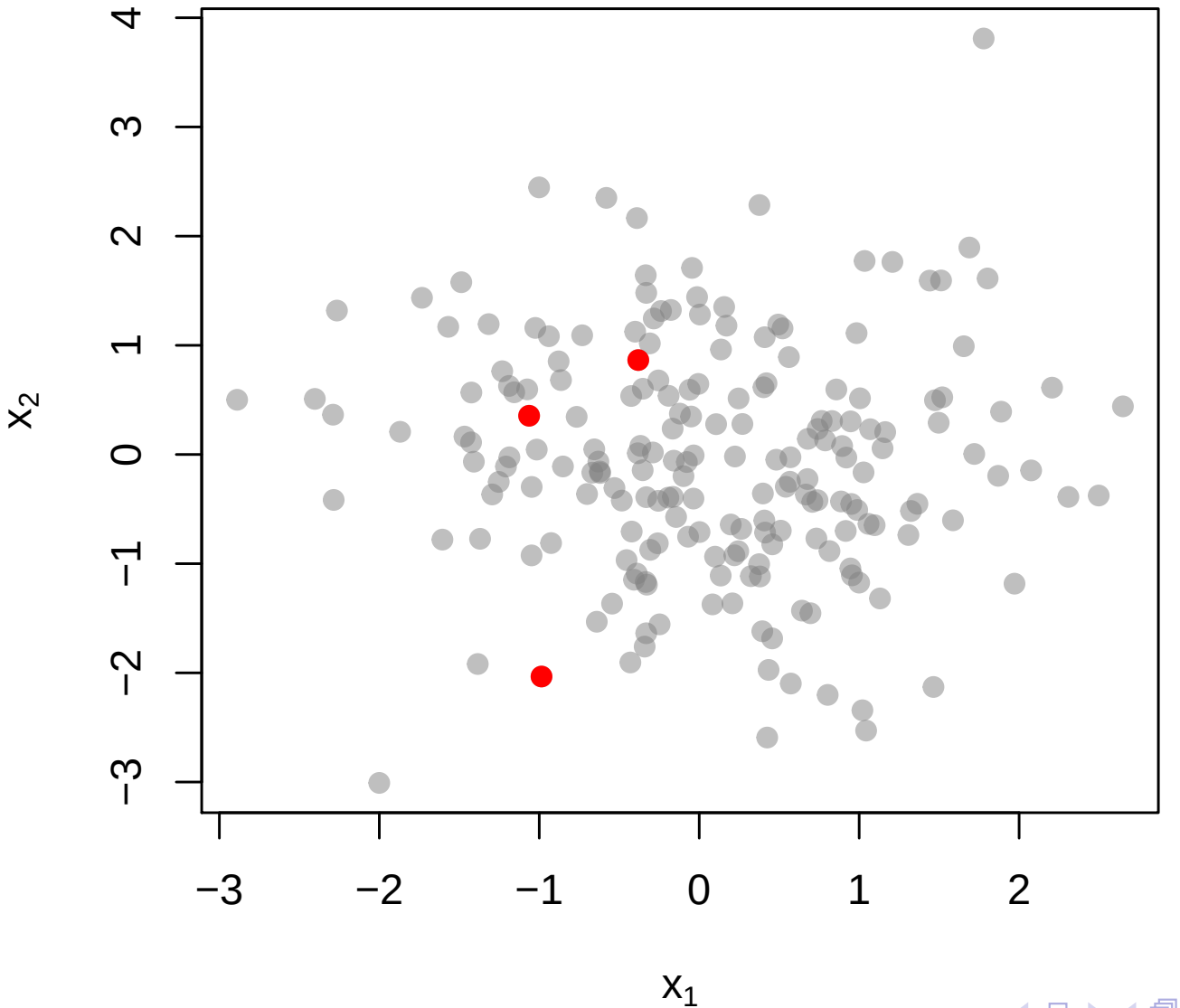
Assumptions

- We have n items, of which we wish to retain k.
- We have a way of quantifying the similarity between items x_i and x_j via a kernel function
- I will start with items that are just points in $\mathbb{R}^2$

Our set



Our set



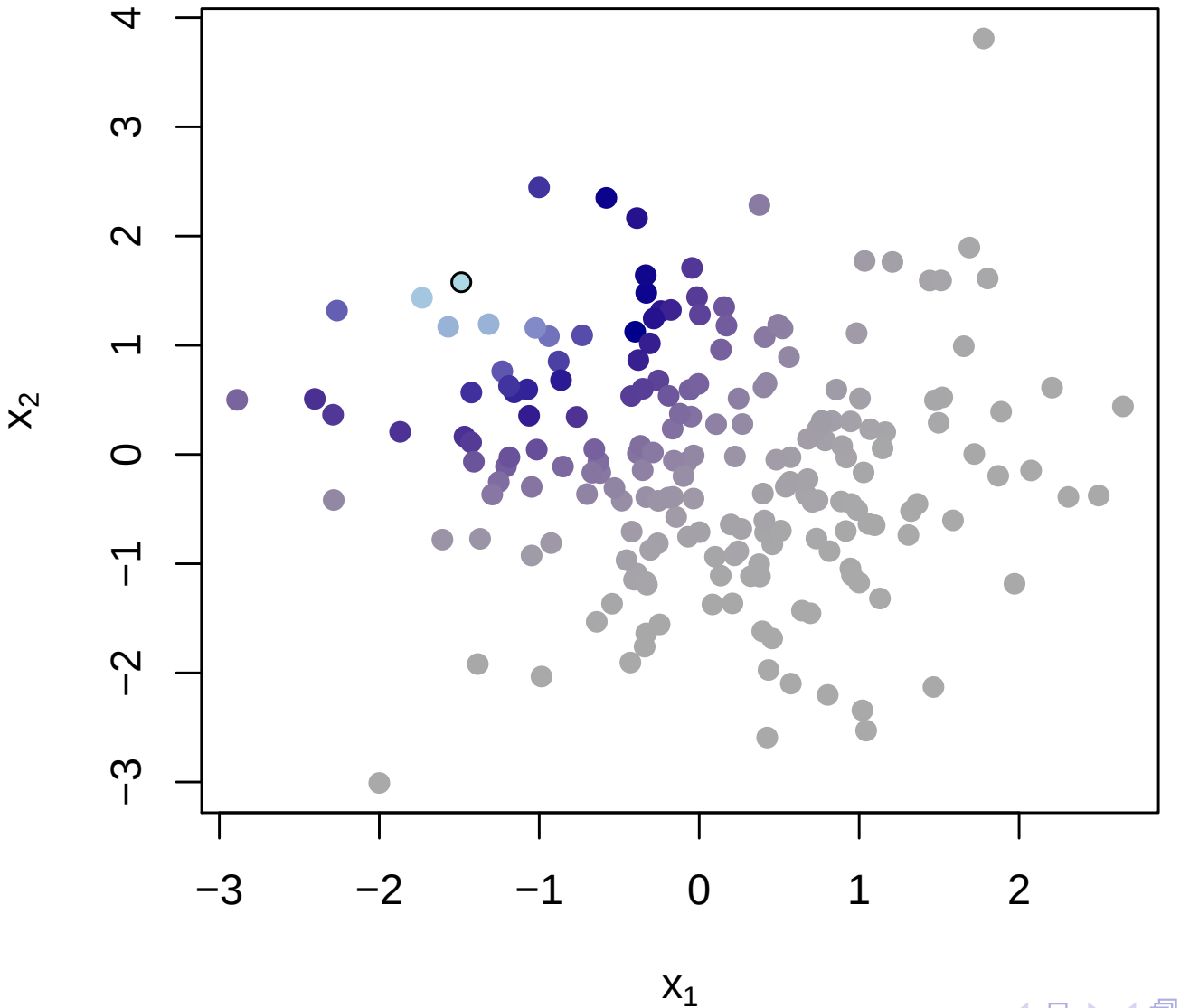
How to enforce diversity: step (1), define similarity

- We need to pick a function that encodes similarity between individuals
- Use any old kernel function you like, e.g.:

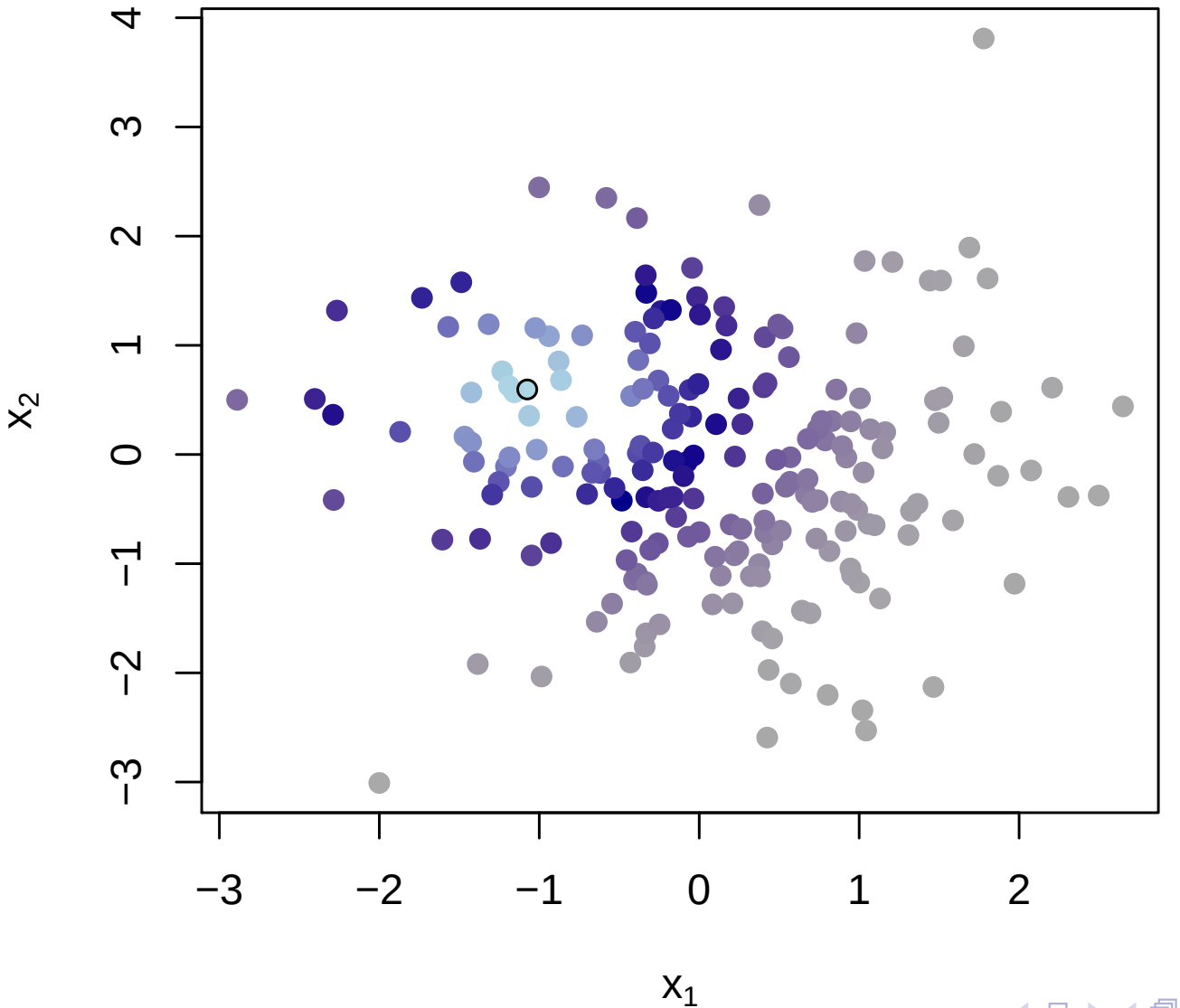
$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{1}{2l^2}||\mathbf{x}_i - \mathbf{x}_j||^2\right)$$

- Similarity is high whenever $\mathbf{x}_i$ is close to $\mathbf{x}_j$

Similarity



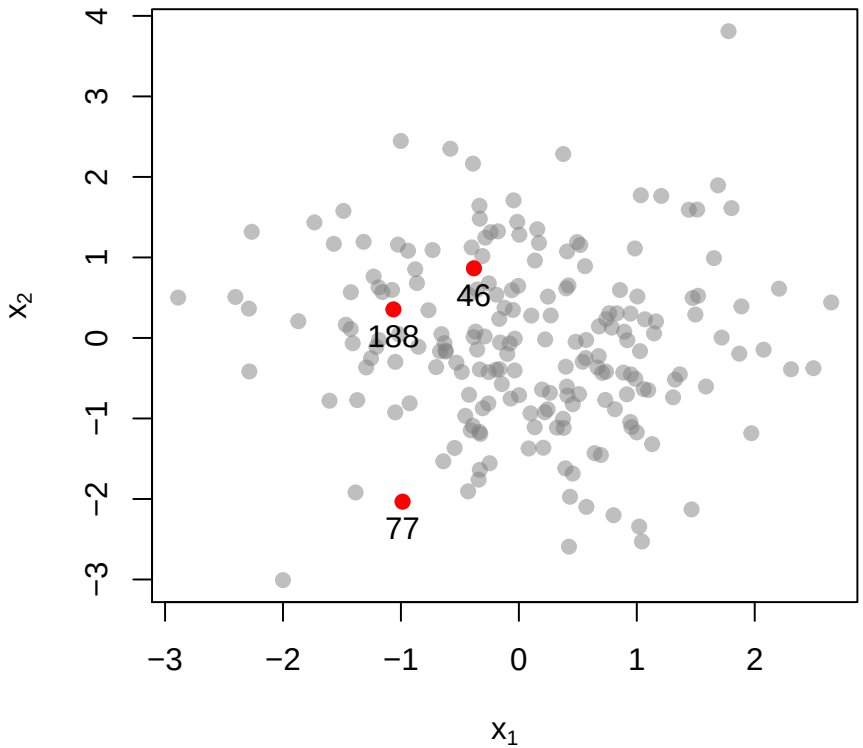
Similarity



How to enforce diversity: step (2), form a matrix

- We encode all pairwise similarities in the set as a matrix, $\mathbf{L}$
- $\mathbf{L}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$
- If we have a subset $\mathcal{X}$, the similarities in the subset are encoded in the matrix $\mathbf{L}_{\mathcal{X}}$, a minor of $\mathbf{L}$
- All matrices $\mathbf{L}_{\mathcal{X}}$ are positive definite.

L-matrices



	46	77	188
46	1.00	0.01	0.70
77	0.01	1.00	0.06
188	0.70	0.06	1.00

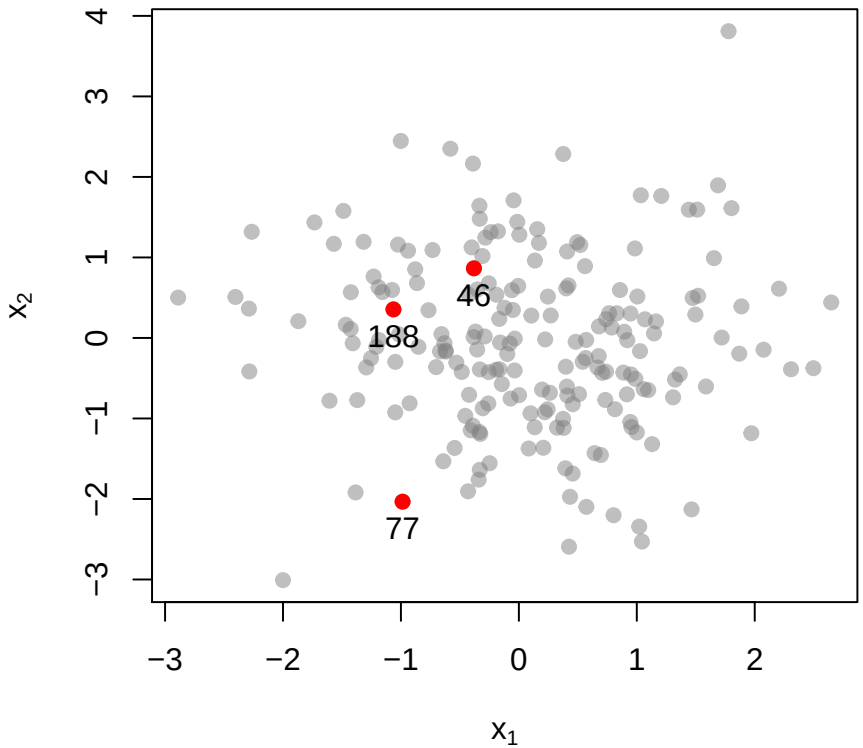
How to enforce diversity: step (3), penalise redundancy

- The final step is to define a probability distribution that penalises redundant sets
- In DPPs, we take:

$$p(\mathcal{X}) \propto \det(\mathbf{L}_{\mathcal{X}})$$

- Sets that contain a lot of similar points will have low $\det(\mathbf{L}_{\mathcal{X}})$, and thus a low probability of being picked.

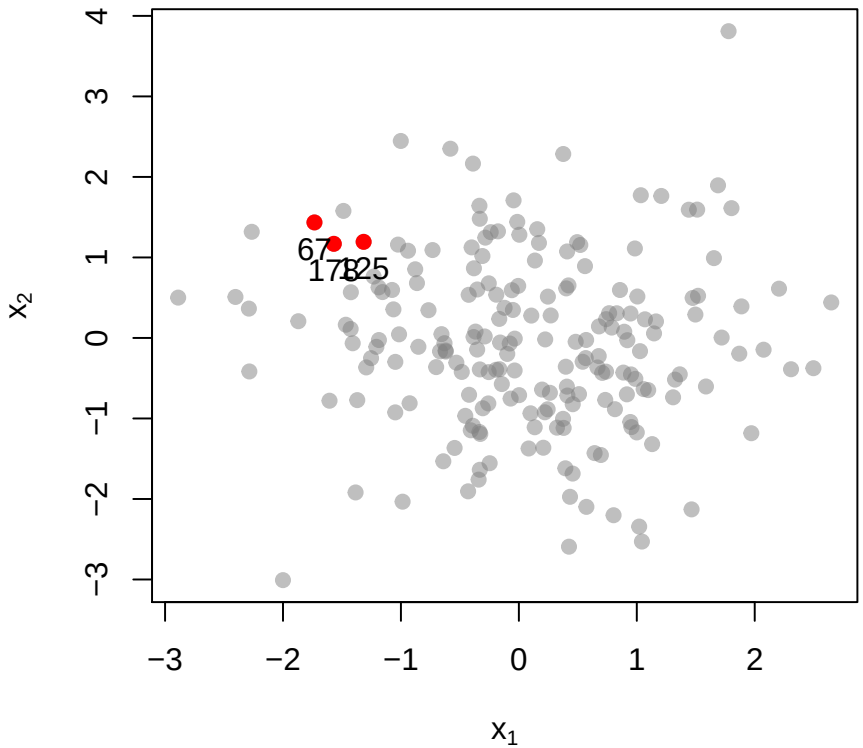
Why does the determinant work?



	46	77	188
46	1.00	0.01	0.70
77	0.01	1.00	0.06
188	0.70	0.06	1.00

Determinant: 0.51.

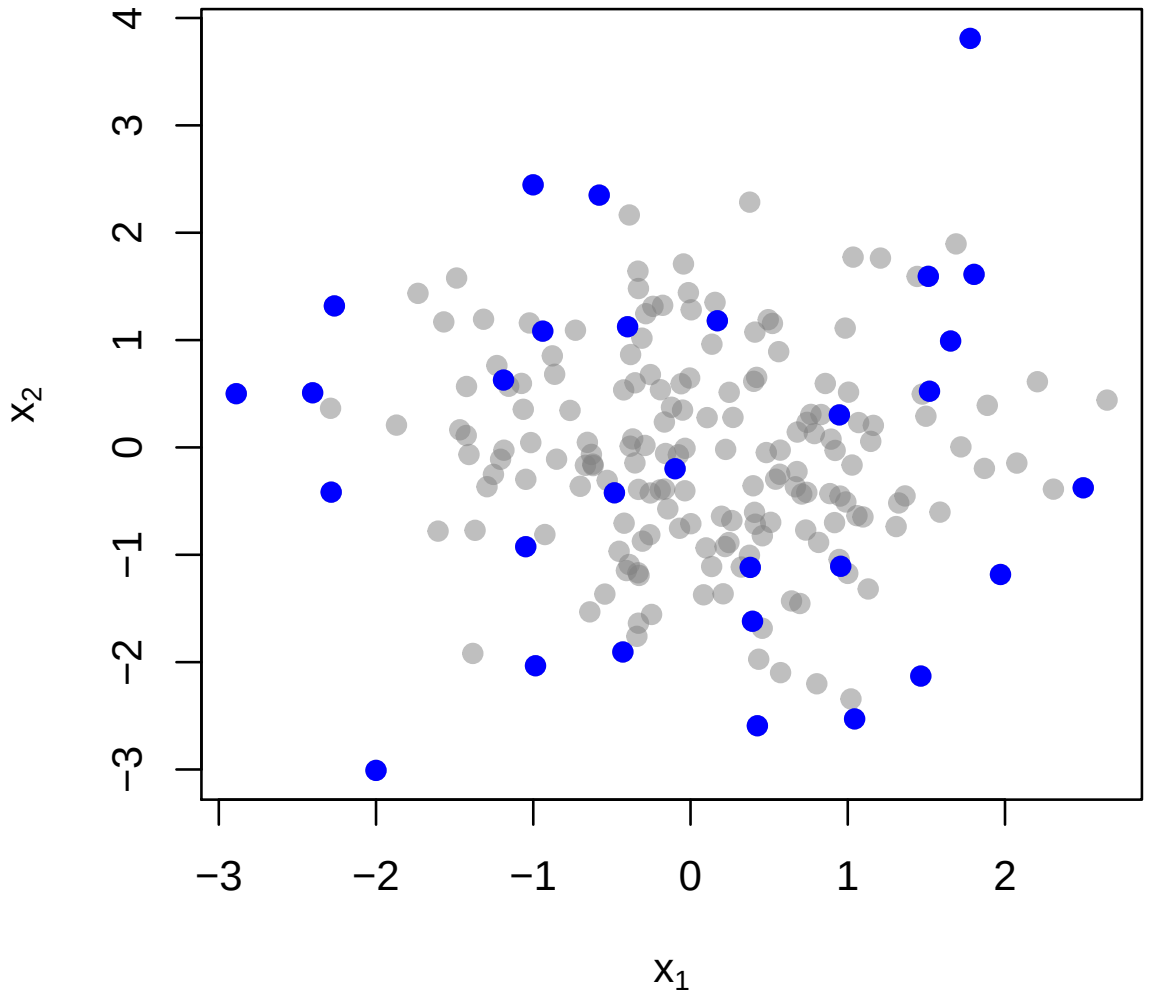
Why does the determinant work?



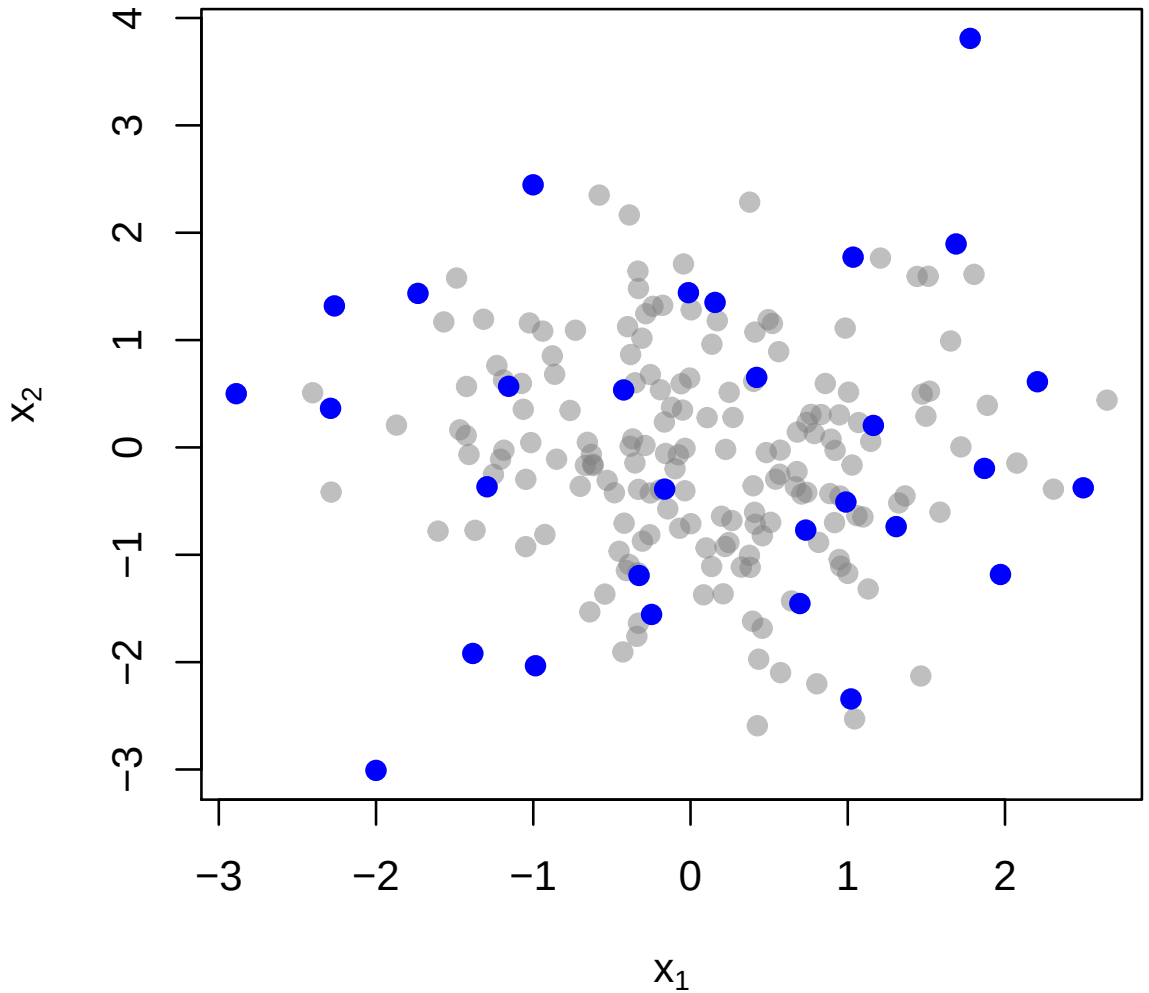
	67	178	125
67	1.00	0.95	0.89
178	0.95	1.00	0.97
125	0.89	0.97	1.00

Determinant: 0.005.

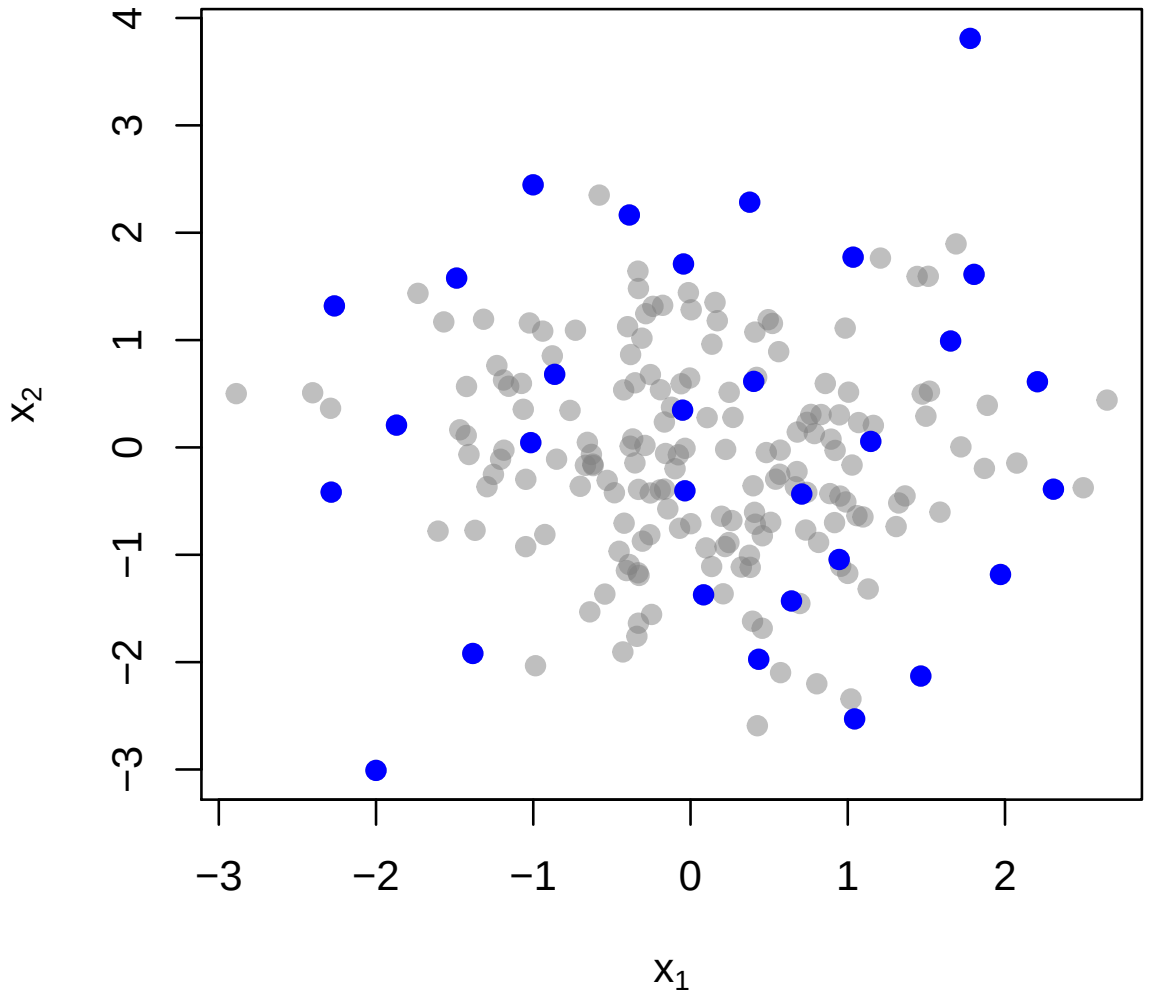
Sampling from a DPP



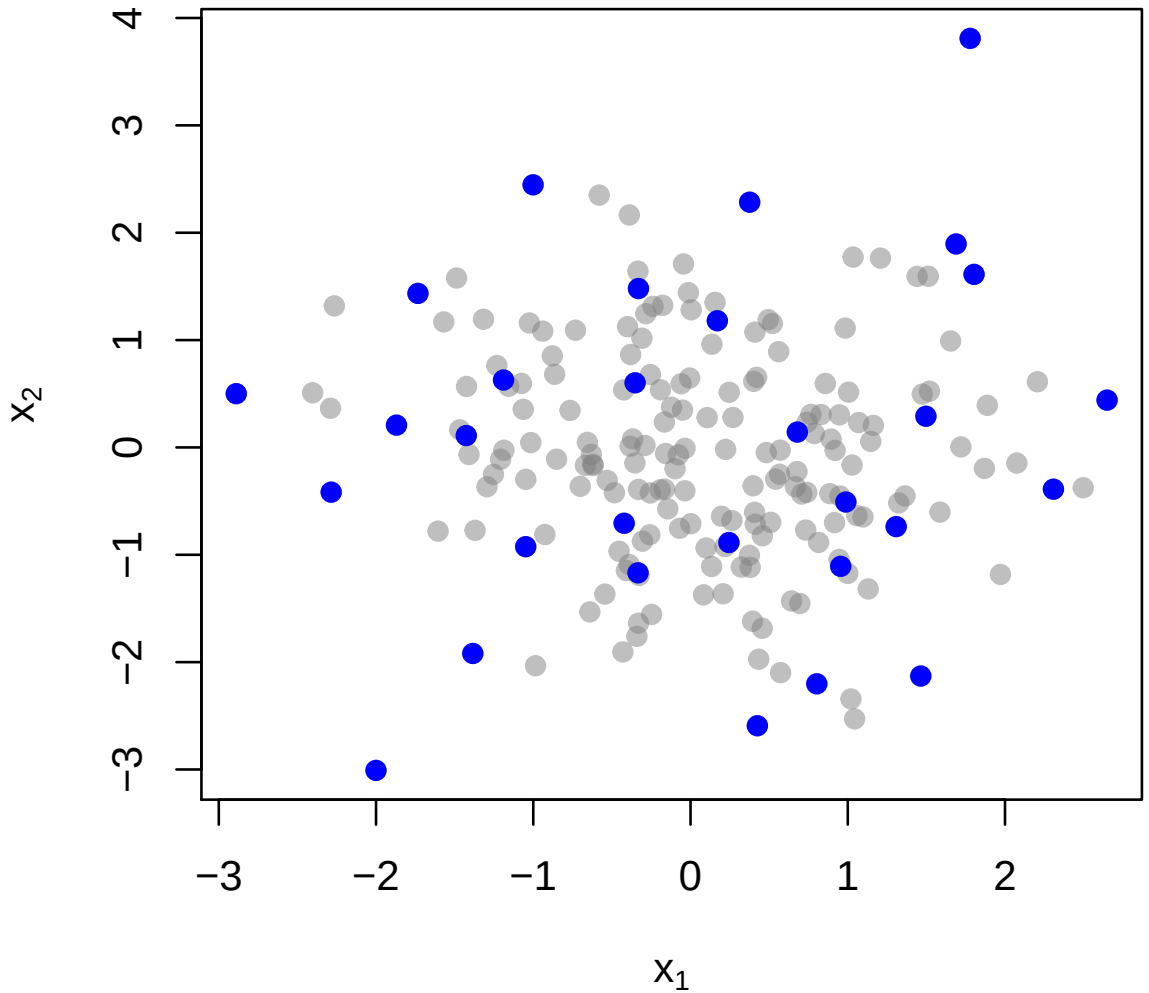
Sampling from a DPP



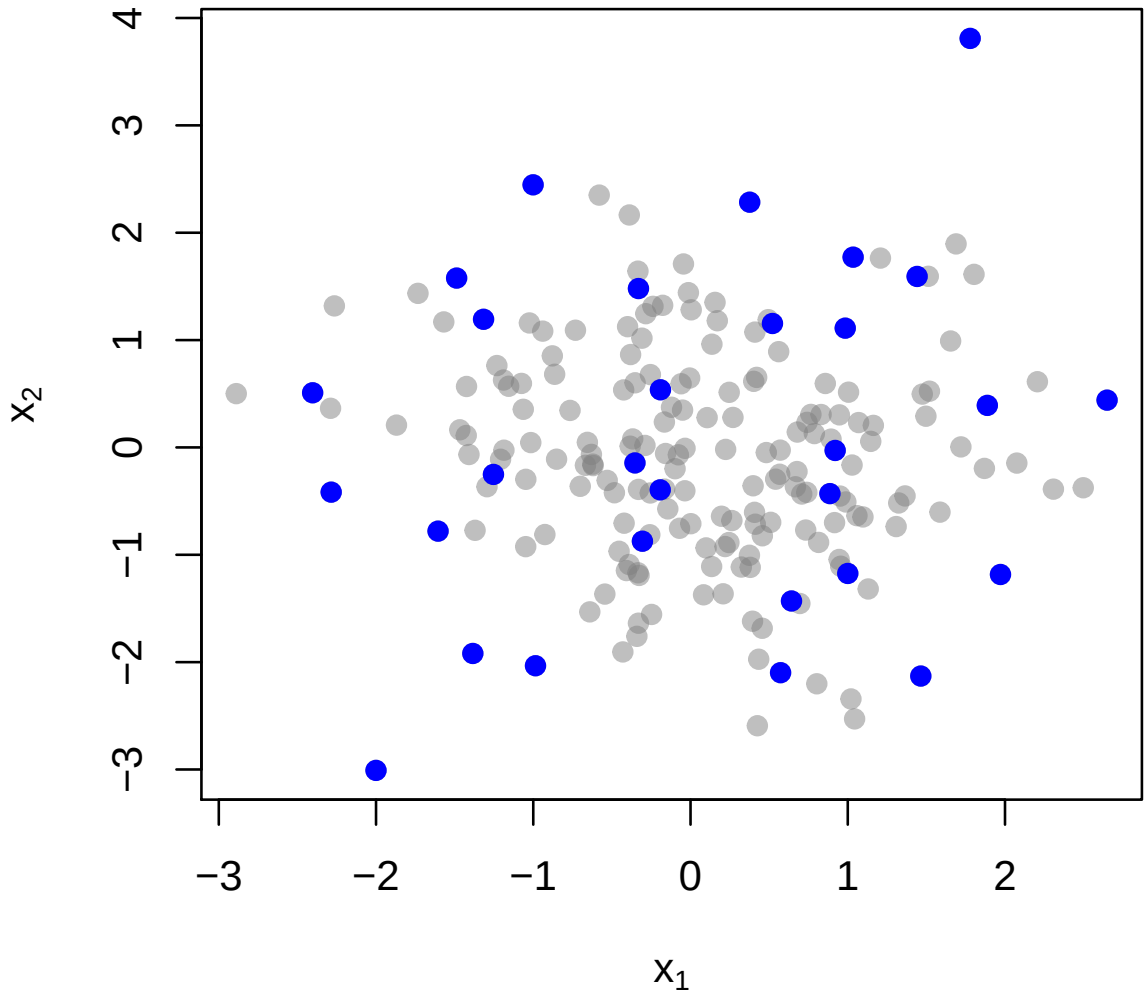
Sampling from a DPP



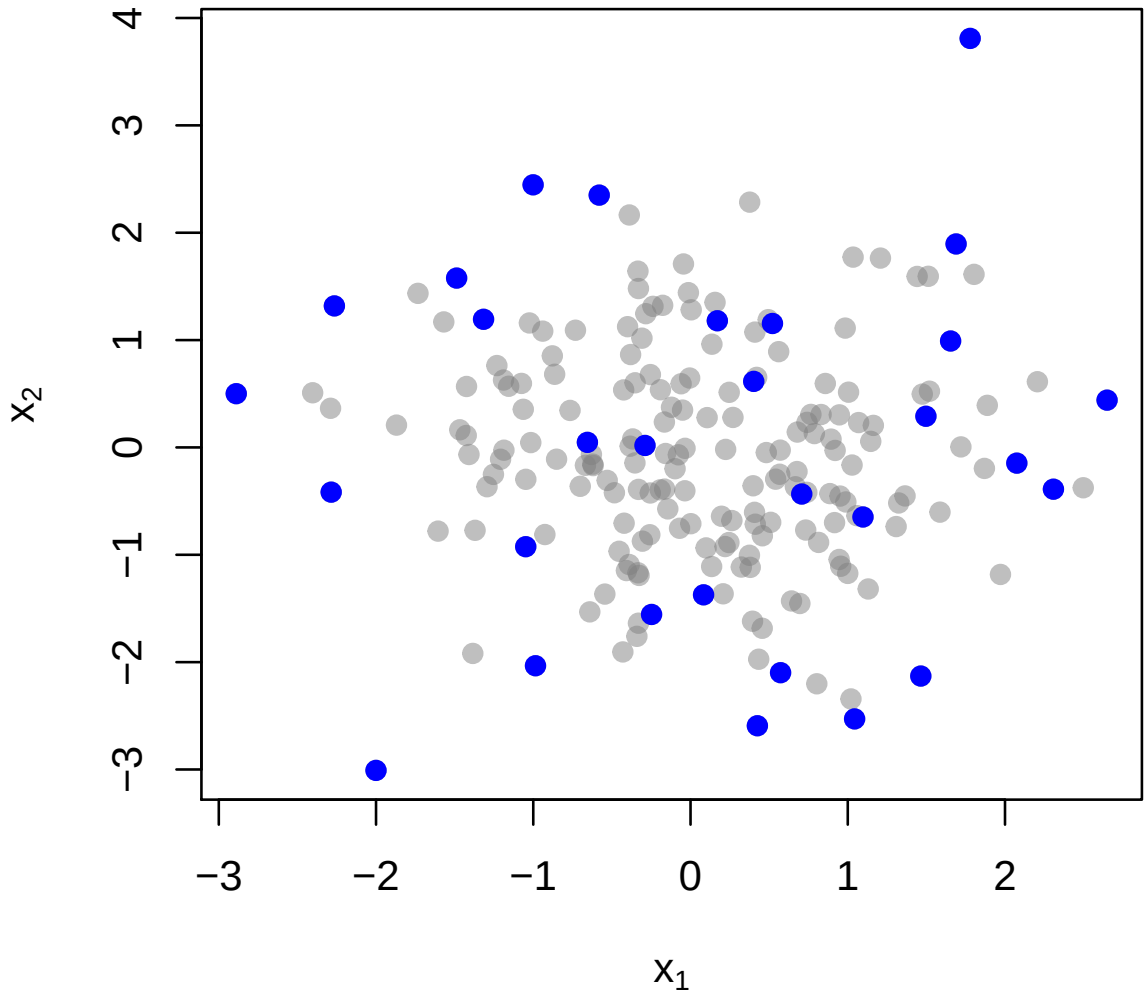
Sampling from a DPP



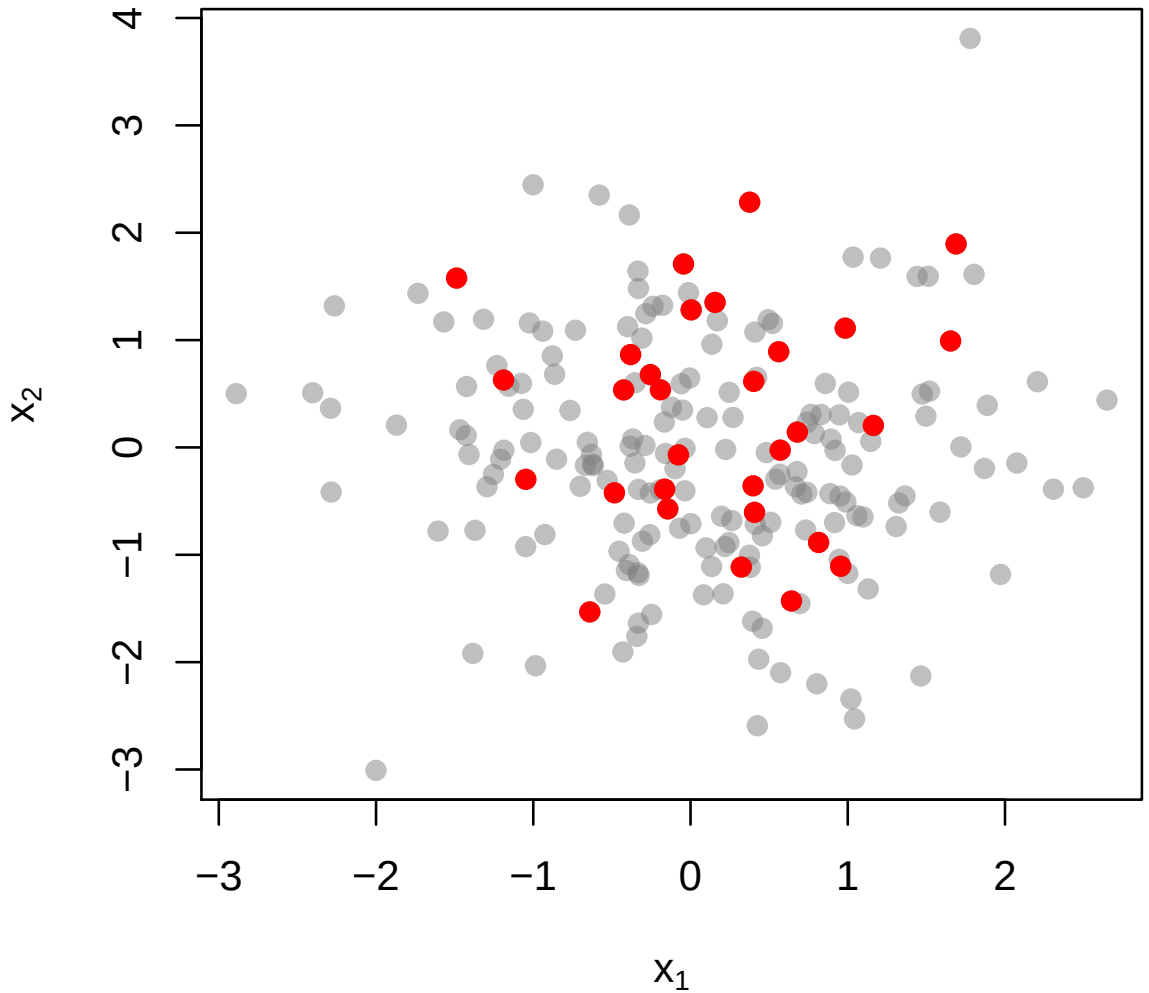
Sampling from a DPP



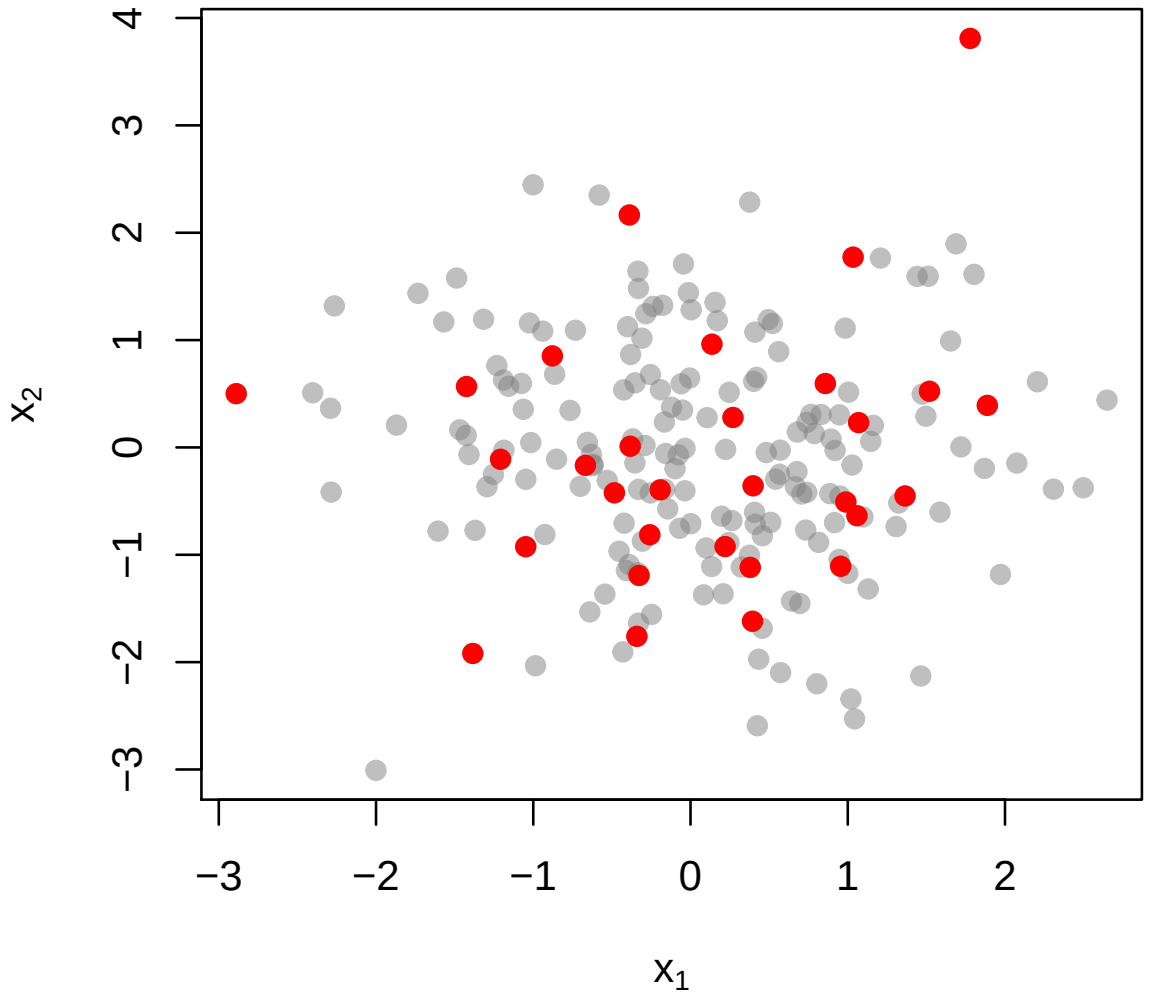
Sampling from a DPP



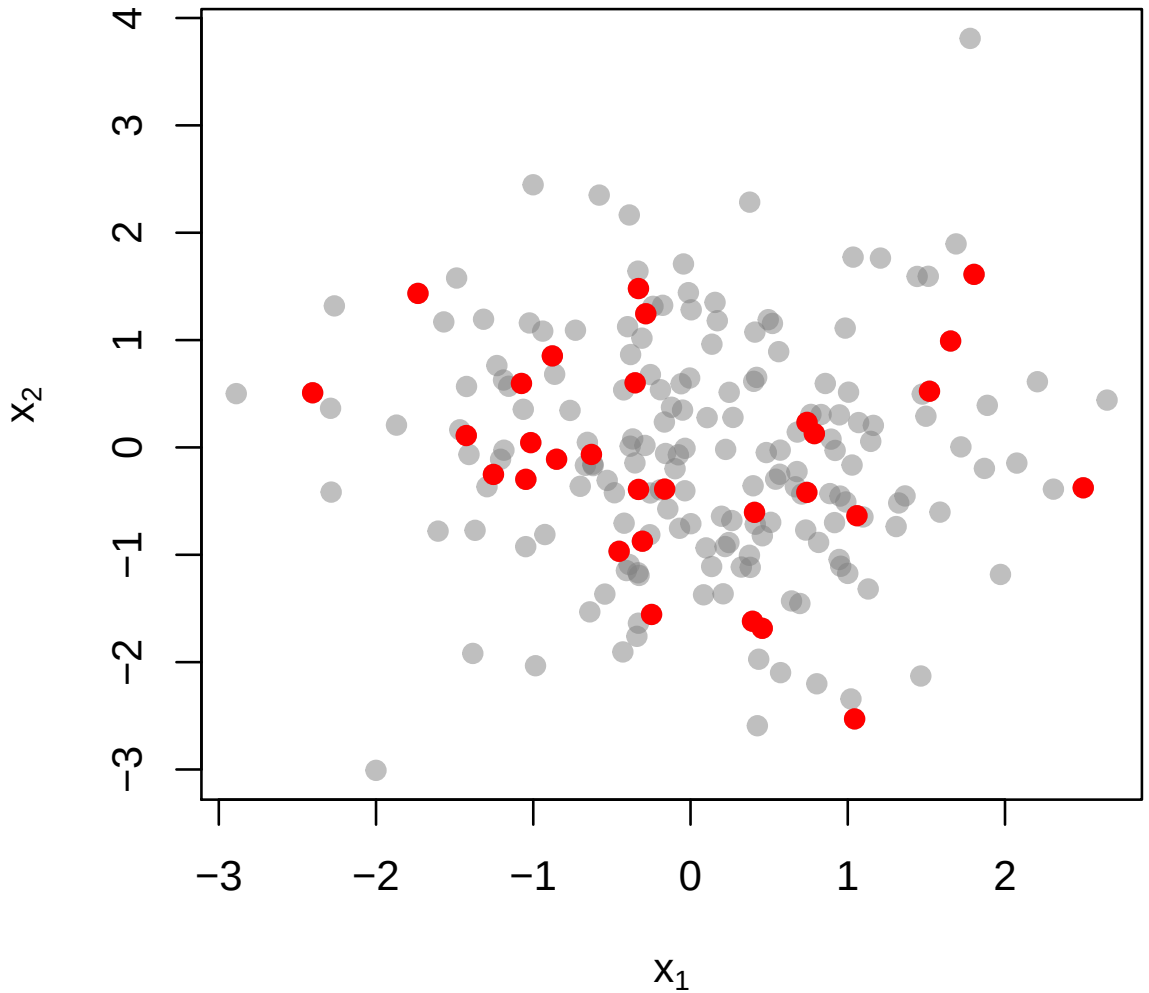
Independent samples



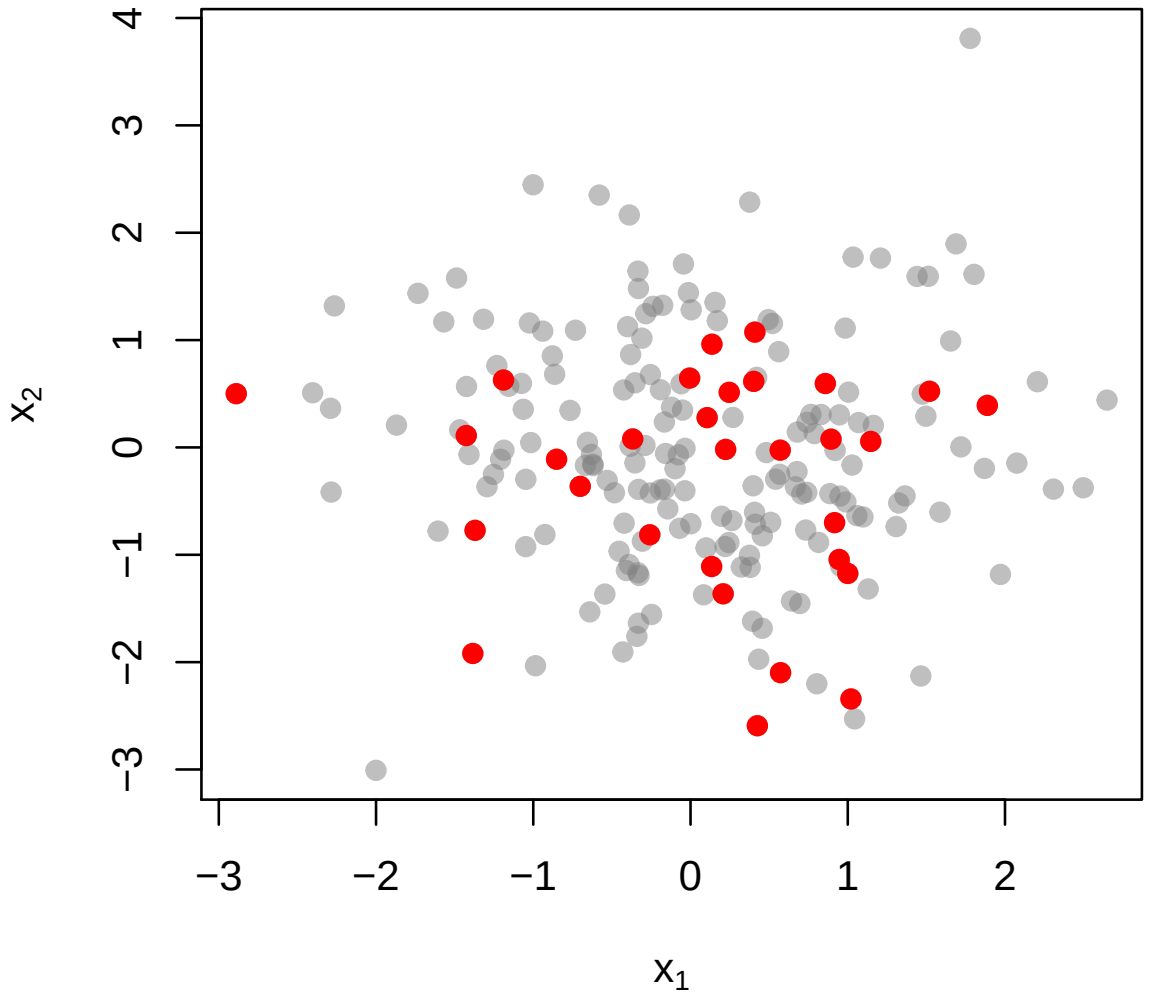
Independent samples



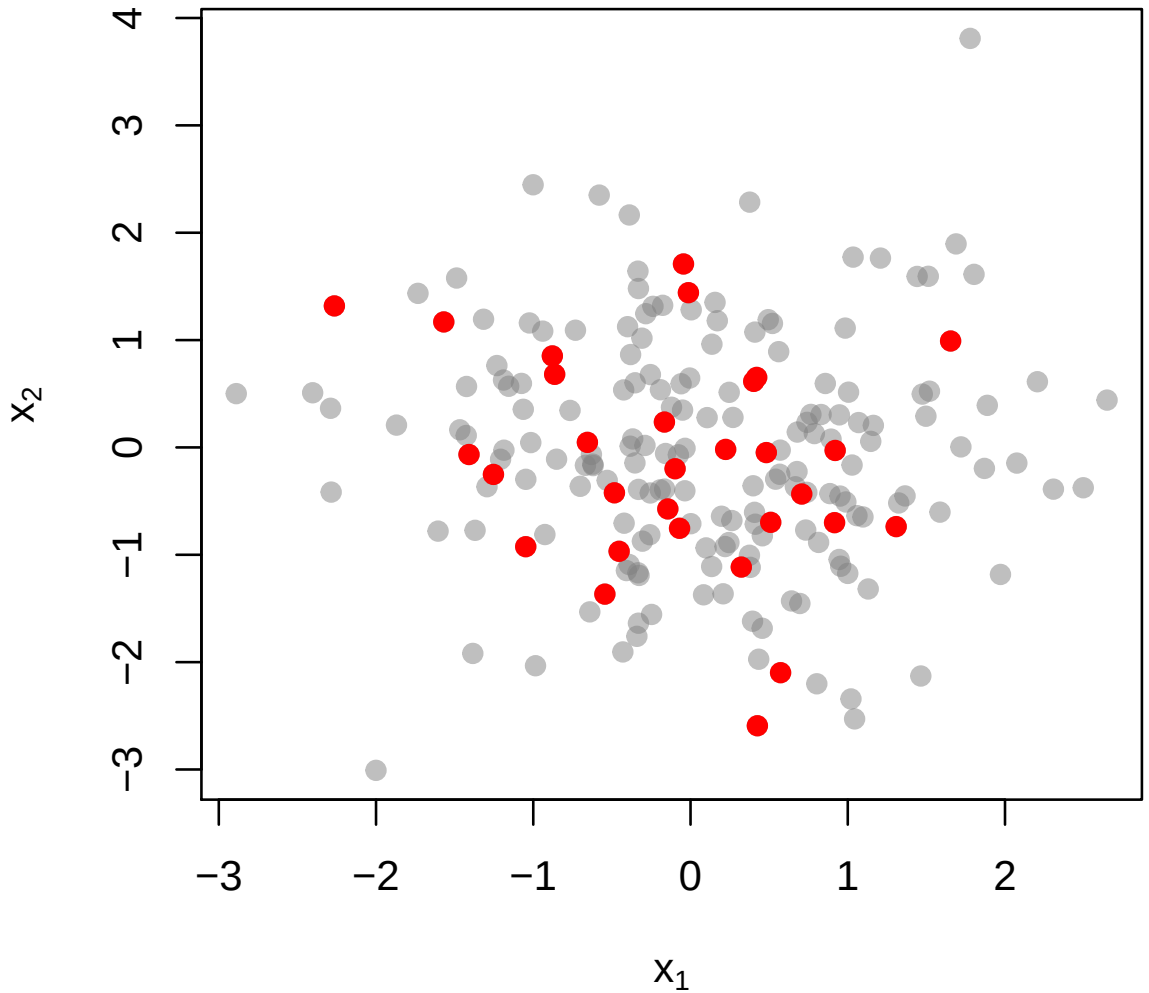
Independent samples



Independent samples



Independent samples



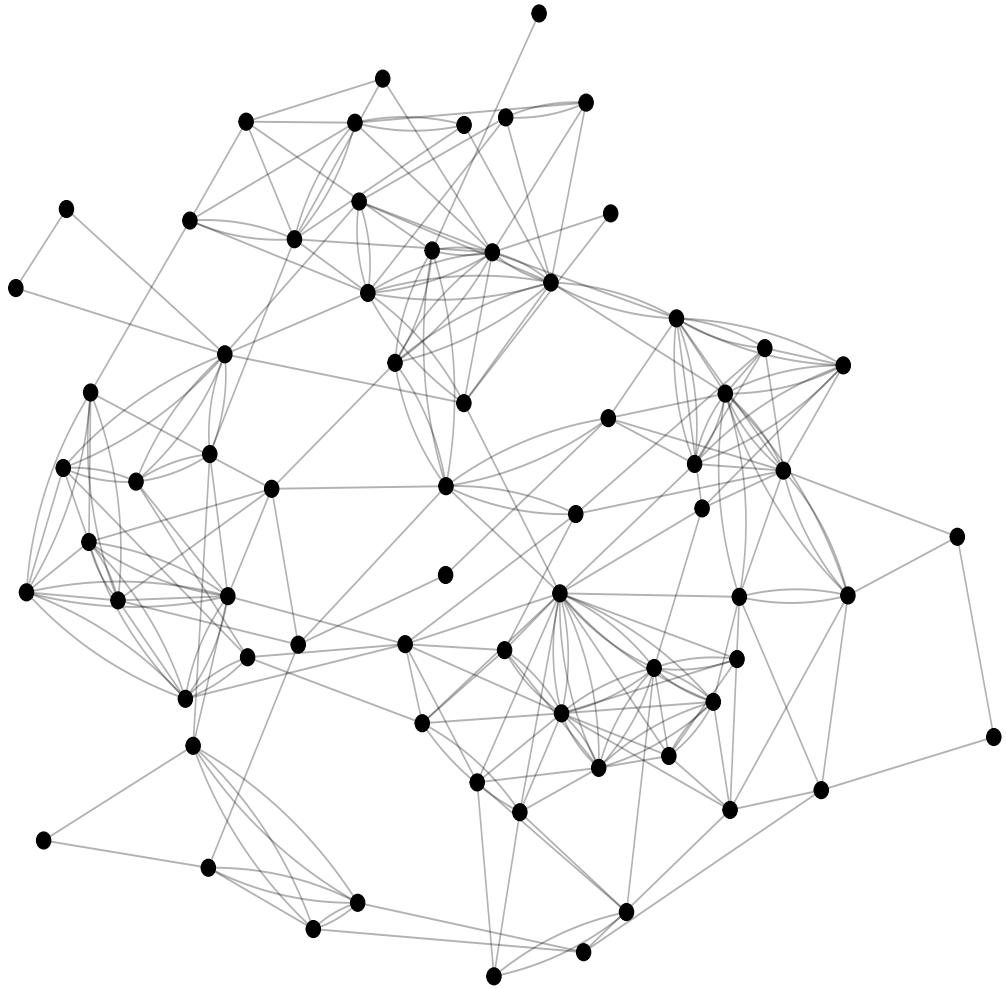
How to sample from a DPP?

- There's a very simple Gibbs sampler (Li et al., 2016)
- Otherwise: for exact sampling, you need the eigendecomposition of a $n \times n$ matrix ($\mathcal{O}(n^3)$ cost. Can be reduced to finding $\mathcal{O}(k)$ top eigenvectors, at $\mathcal{O}(nk^2)$ cost.
- Then the sampling algorithm runs in $\mathcal{O}(nk^2)$ (don't use the one in Kulesza and Taskar (2012), it's outdated, see e.g. Tremblay et al. (2017b); Barthelmé et al. (2018))

In what sense are DPPs tractable?

- It's rare to see a point process that has all of following:
  1. Joint density is tractable
  2. Inclusion probabilities are tractable, i.e. you can compute the prob. that item i is in the random set $\mathcal{X}$, or that i, j are jointly in $\mathcal{X}$, etc.
  3. Sampling is tractable
- DPPs have all three features (with caveats), meaning you can actually prove stuff
- (doesn't mean they're always the best point process!)

Subsampling a graph



"Highschool graph" (Coleman, 1964)

Subsampling a graph

- Tremblay et al. (2017a,b): consider a graph with n nodes
- Each node has x_i an associated *signal* y_i
- Goal: measure f on a limited subset of nodes, such as to reconstruct the signal $\mathbf{y}$ on the missing nodes
- The signal is assumed to be smooth over the graph

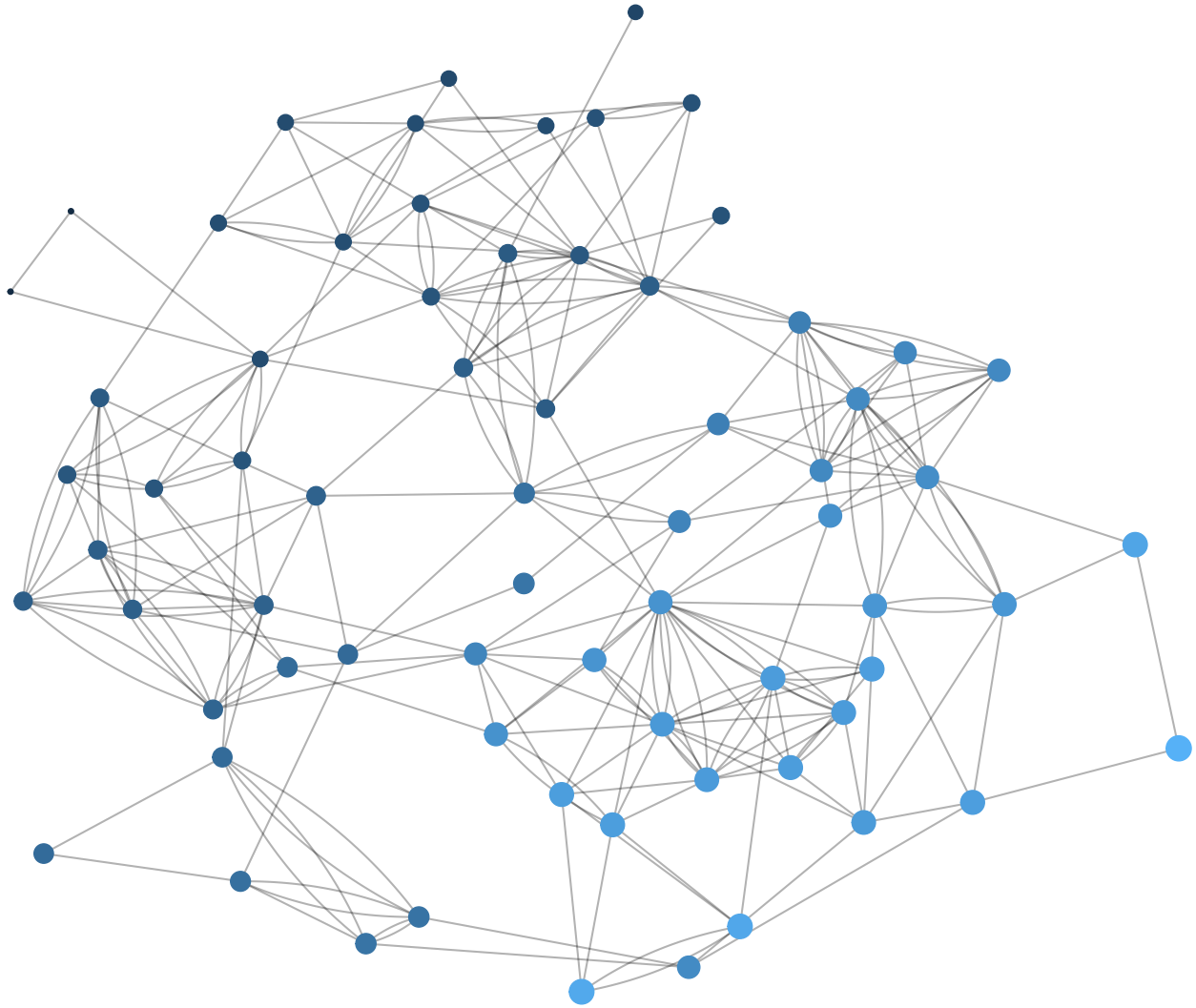
Smooth signal on a graph

- A smooth signal on a graph roughly means that neighbouring nodes are likely to have similar values
- In the field of graph signal processing, the notion is made precise via the graph Laplacian

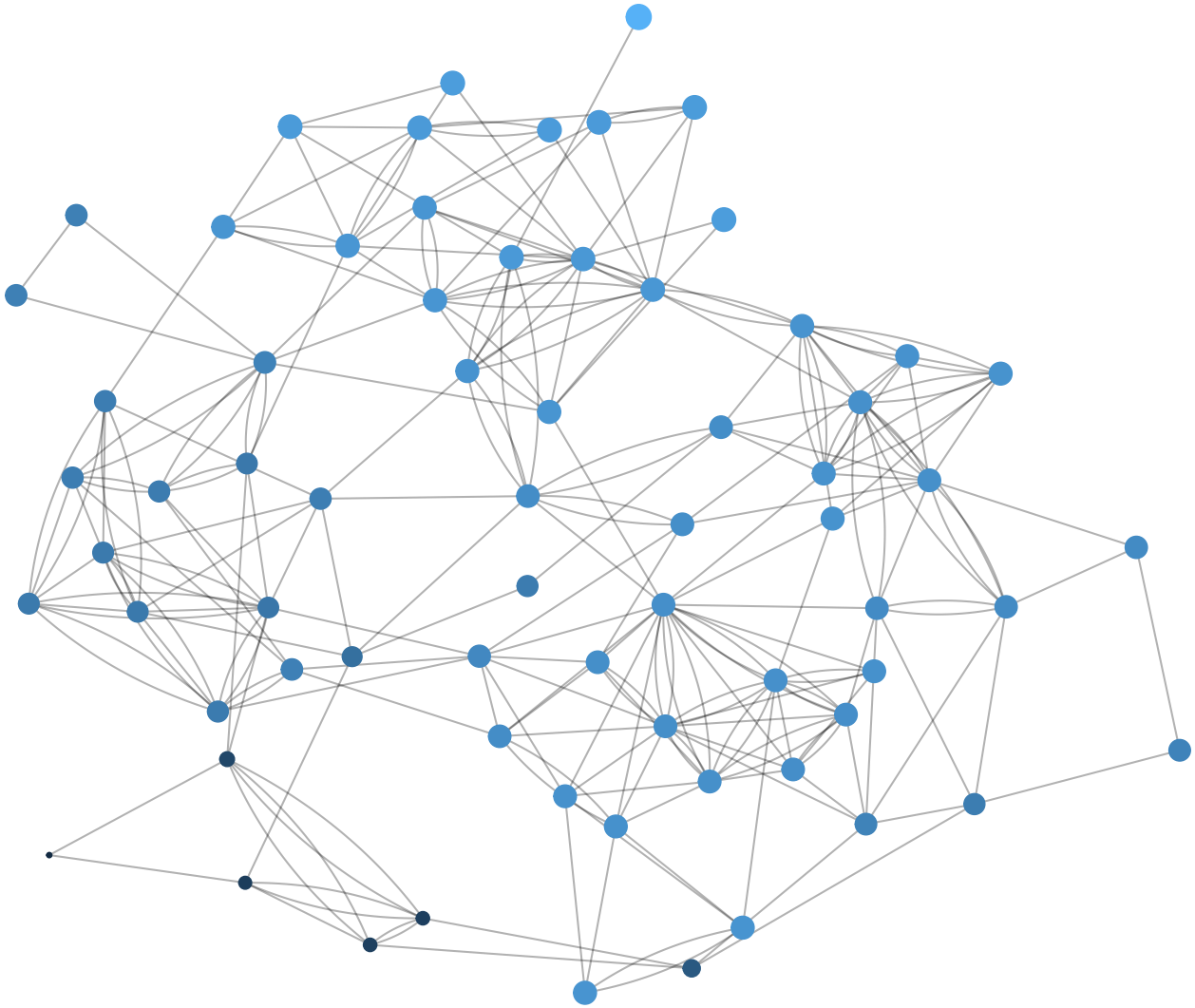
$$\mathbf{y} = \sum_{j=1}^{m} \alpha_j \mathbf{u}_j$$

- Signal lies in the span of the first m eigenvectors of the graph Laplacian $\mathbf{u}_1, \ldots, \mathbf{u}_m$
- If graph is a grid, same thing as a band-limited signal in the traditional sense
- If graph has m separate communities, value is constant within a community.

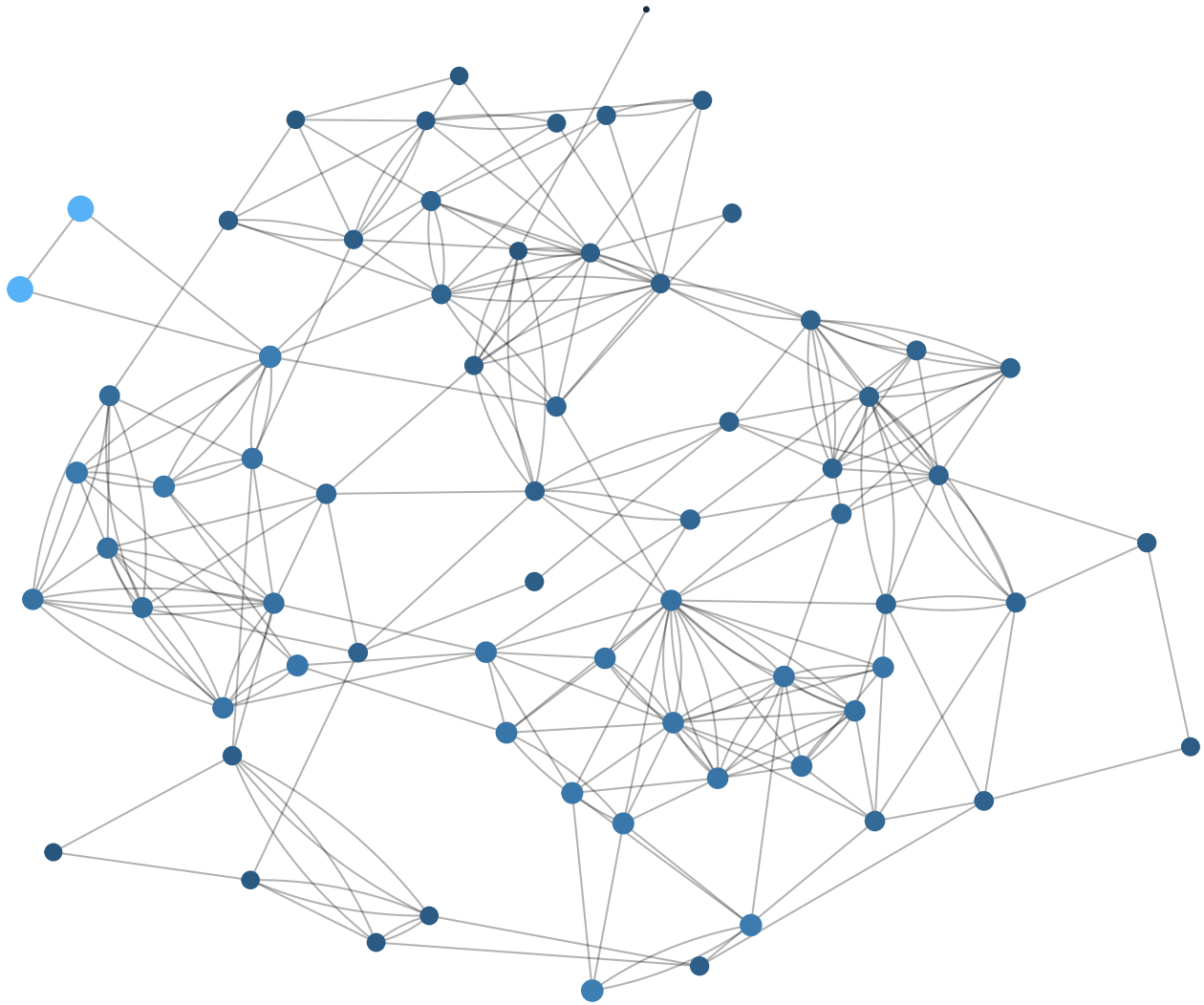
Eigenvectors of the graph Laplacian



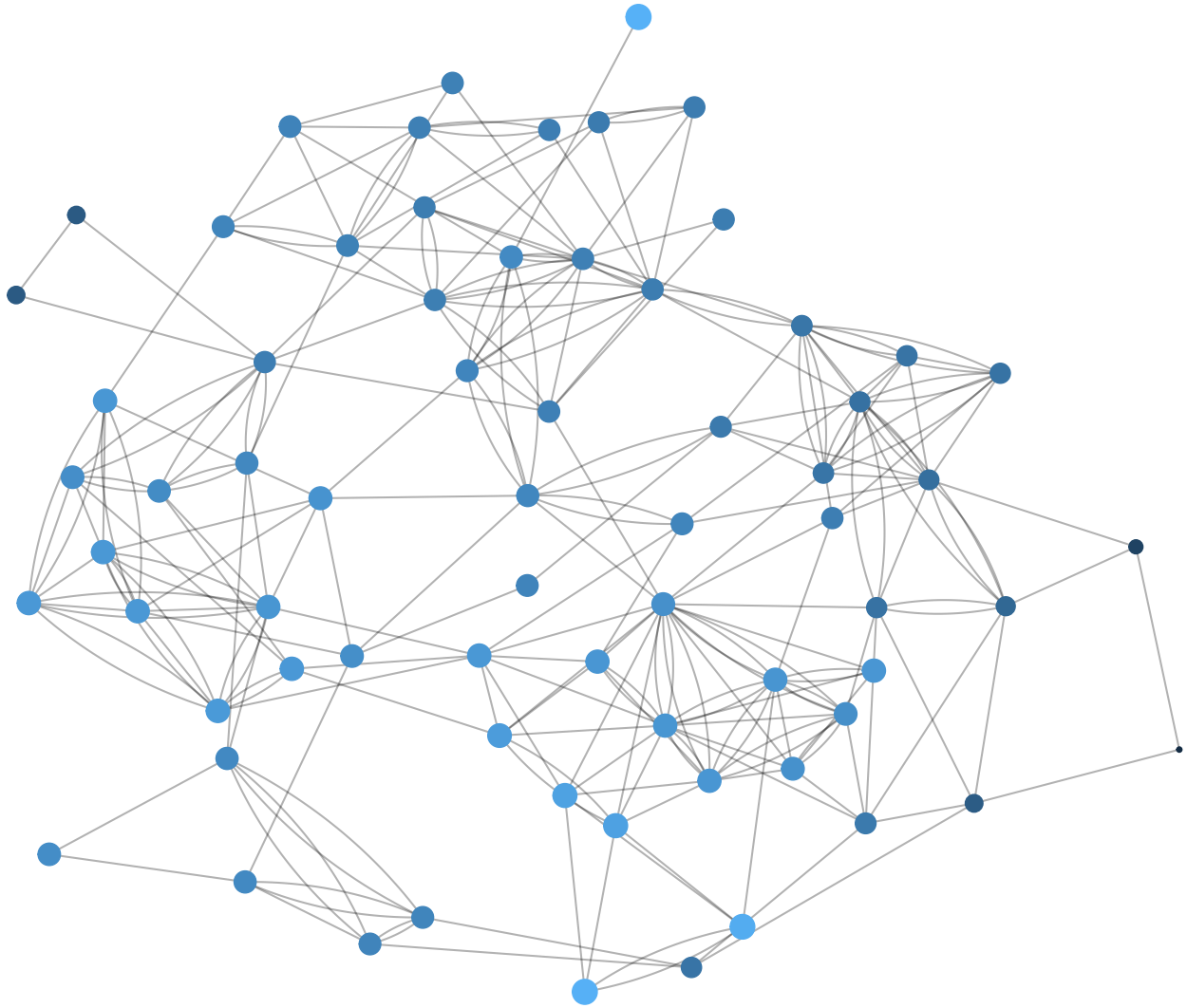
Eigenvectors of the graph Laplacian



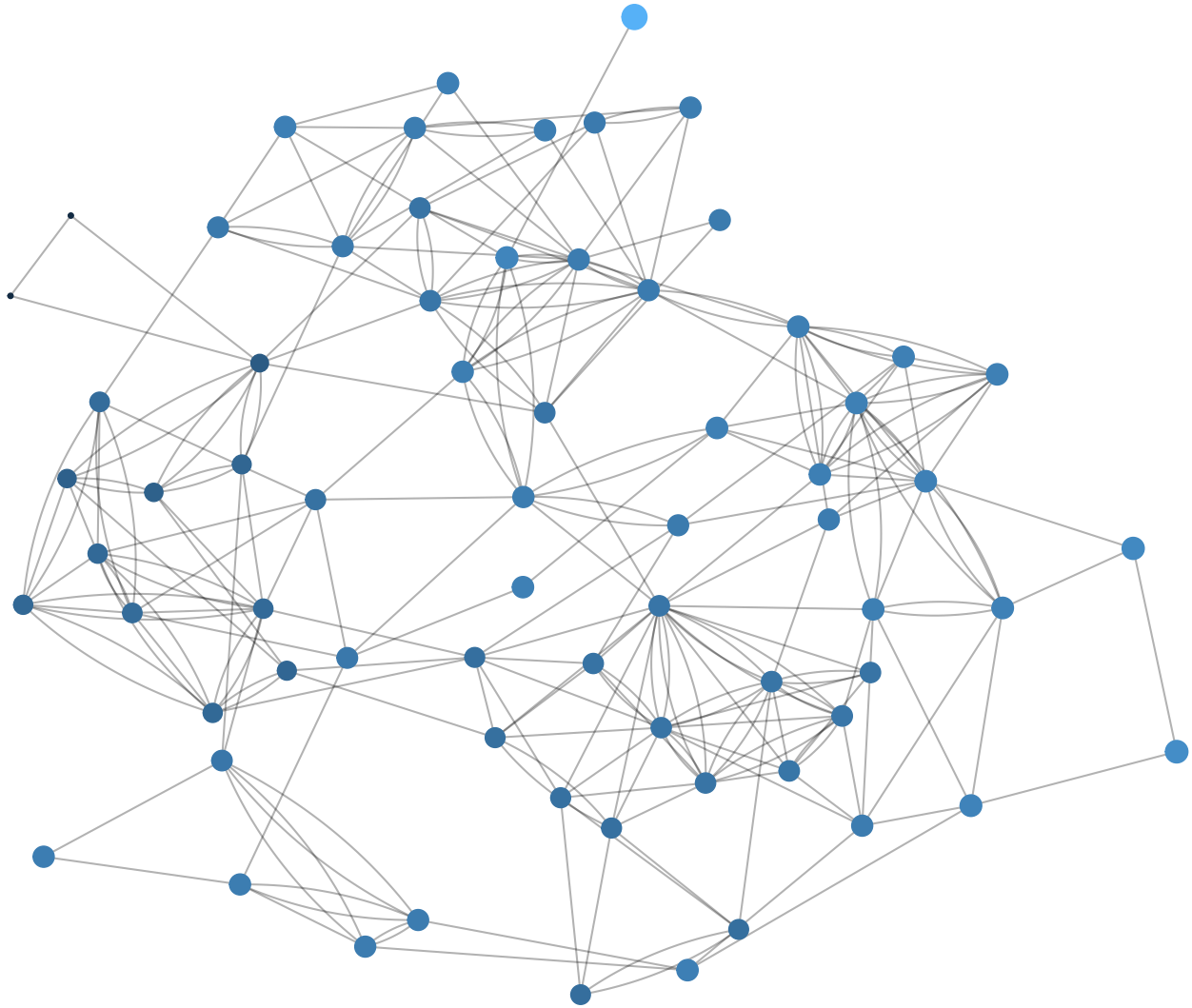
Eigenvectors of the graph Laplacian



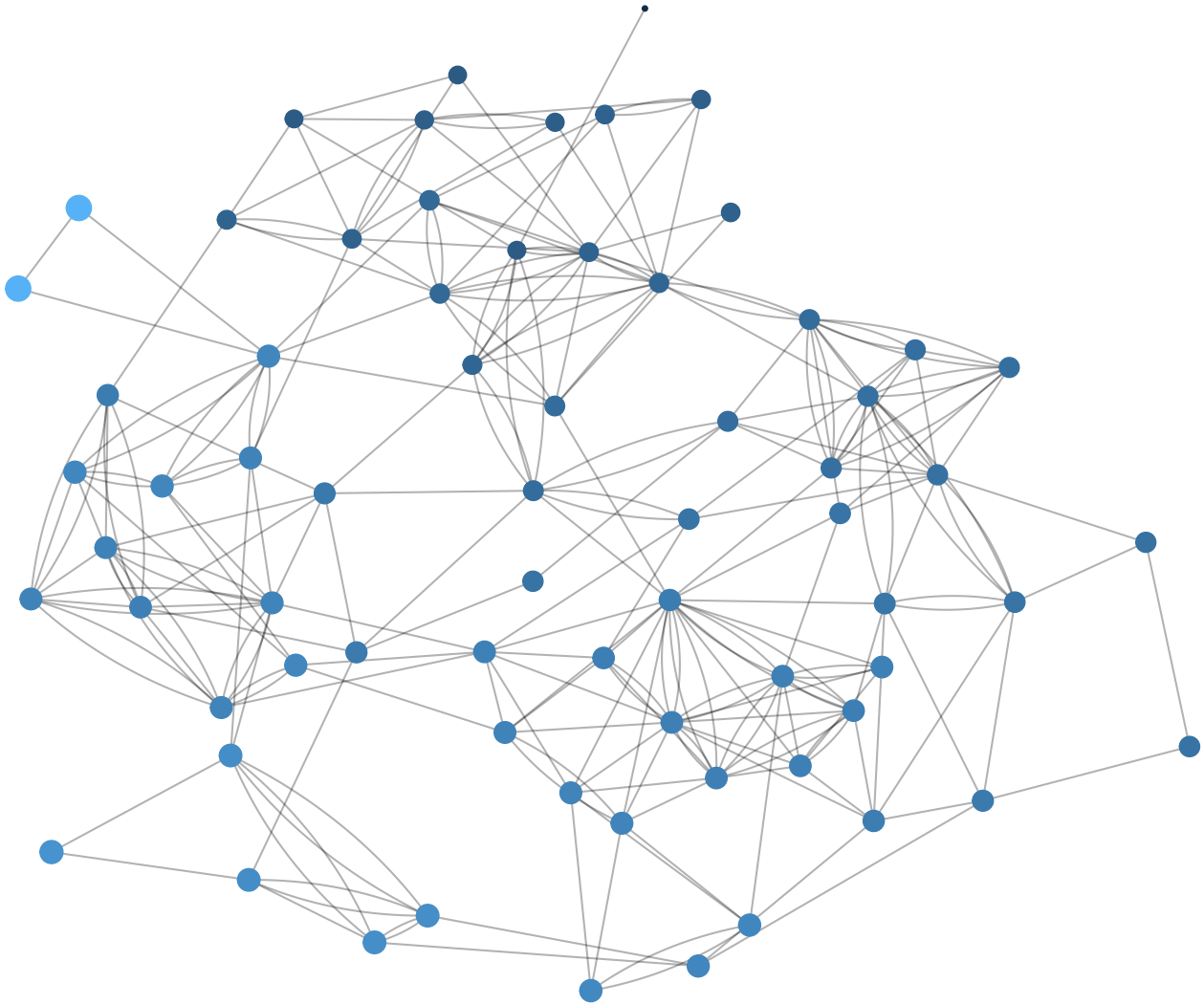
Eigenvectors of the graph Laplacian



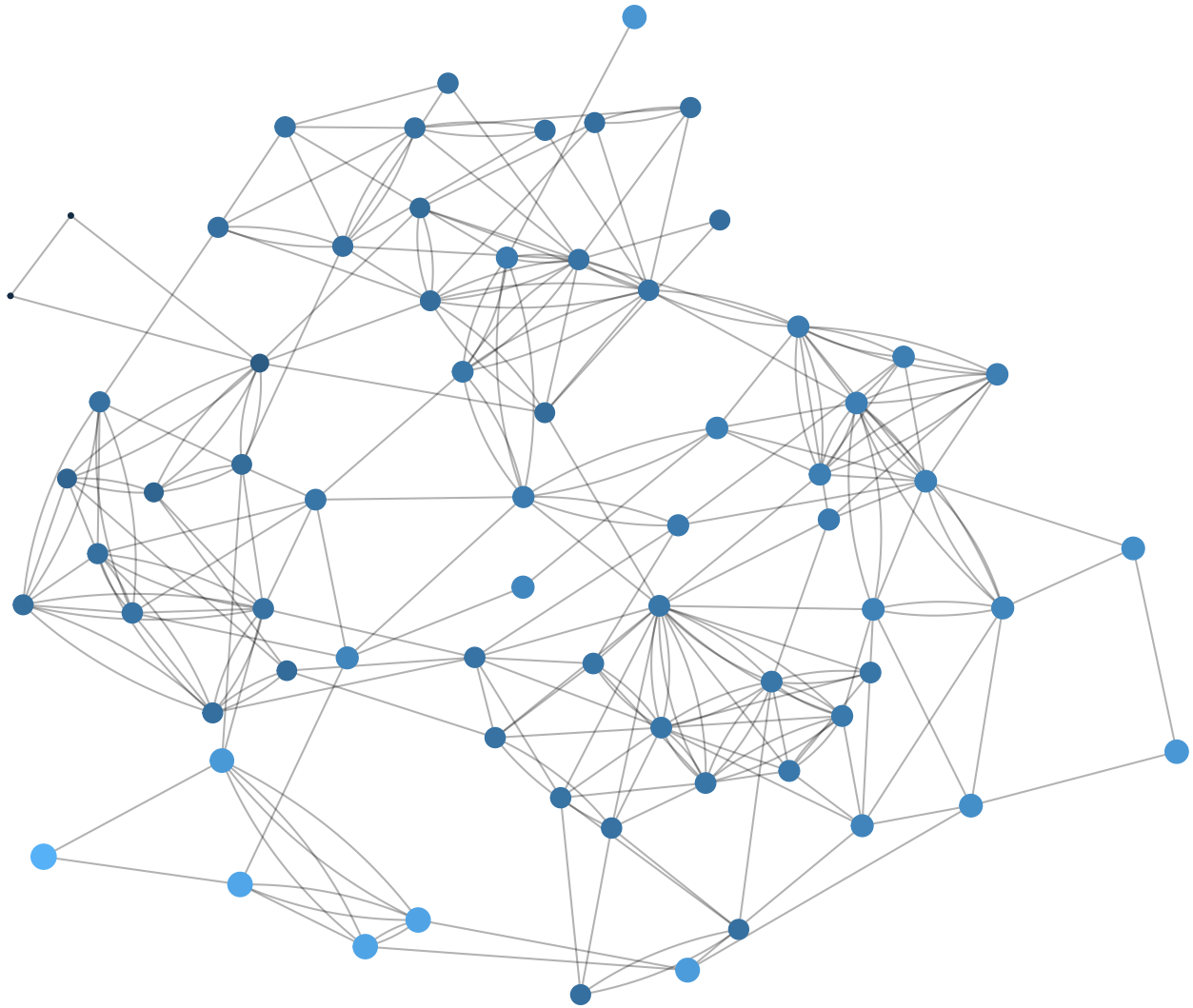
Random signals



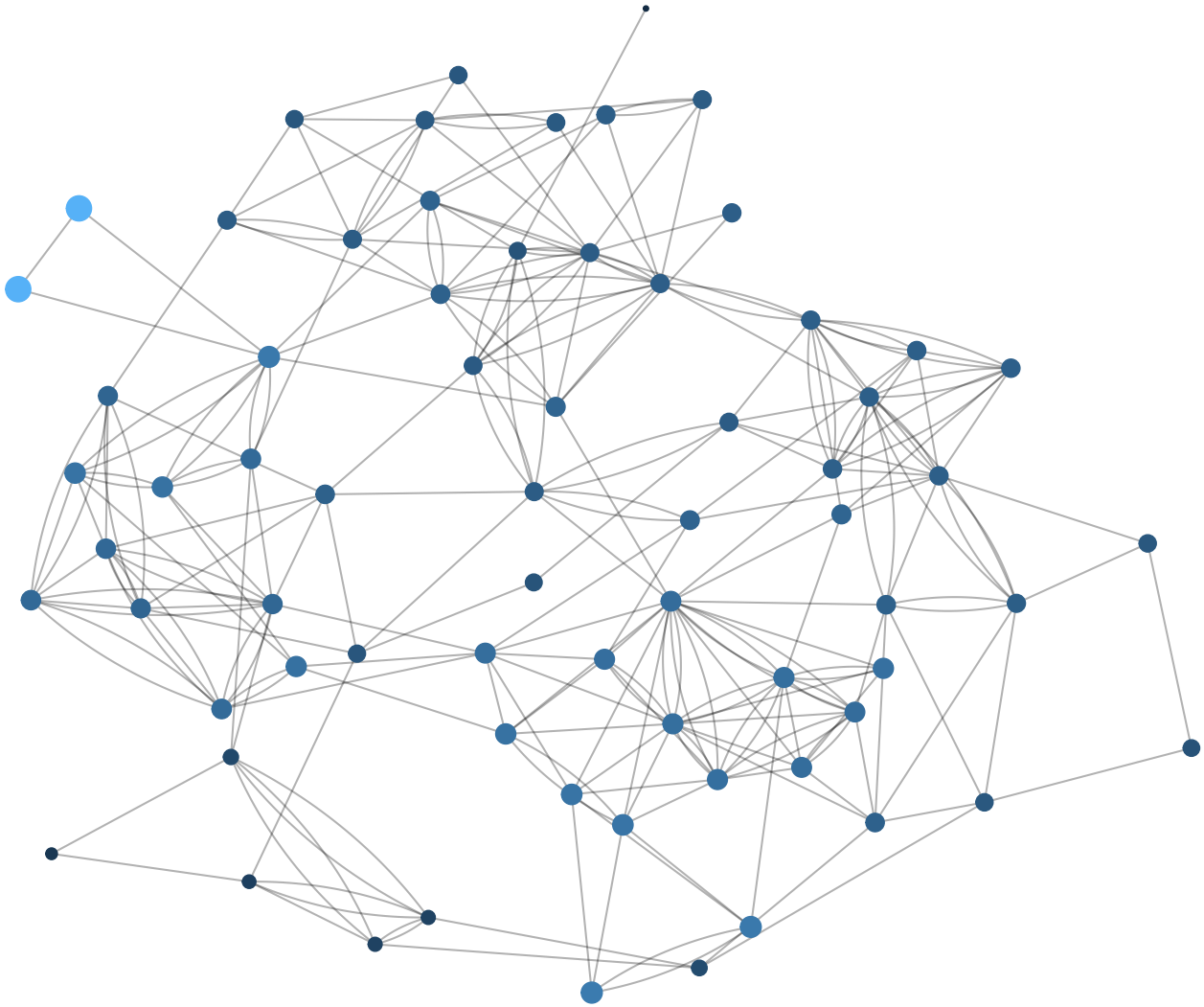
Random signals



Random signals



Random signals



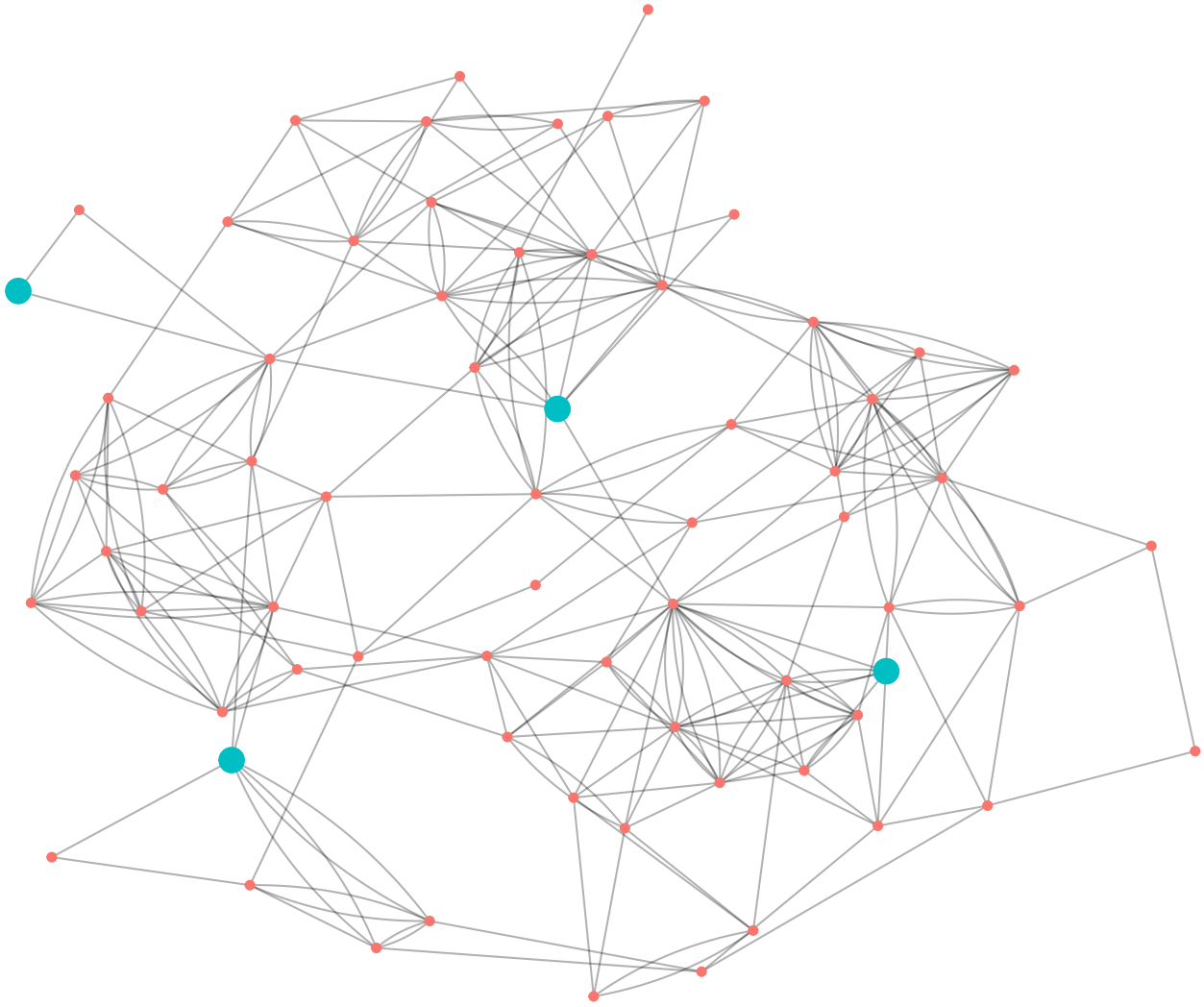
Using a DPP to sample nodes in a graph

- We can use the following kernel for a DPP:

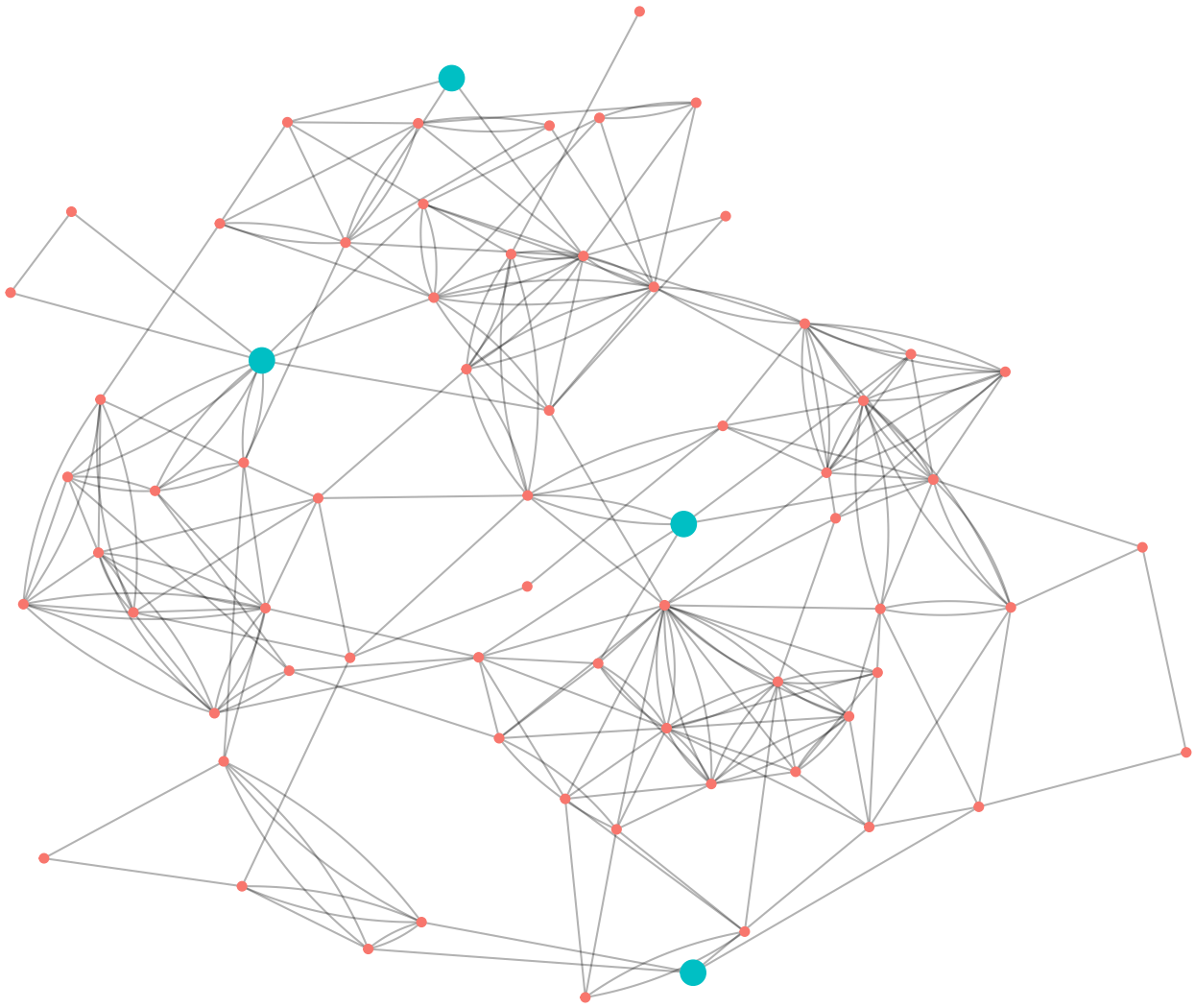
$$\mathbf{L} = \mathbf{U}\mathbf{U}^t$$

- Here $\mathbf{U}$ contains the first m eigenvectors of the graph
- Guarantees perfect reconstruction, because $\mathbf{U}_{\mathcal{X},:}$ is invertible (otherwise the determinant of $\mathbf{U}_{\mathcal{X},:}\mathbf{U}^t_{:,\mathcal{X}}$ would be 0).

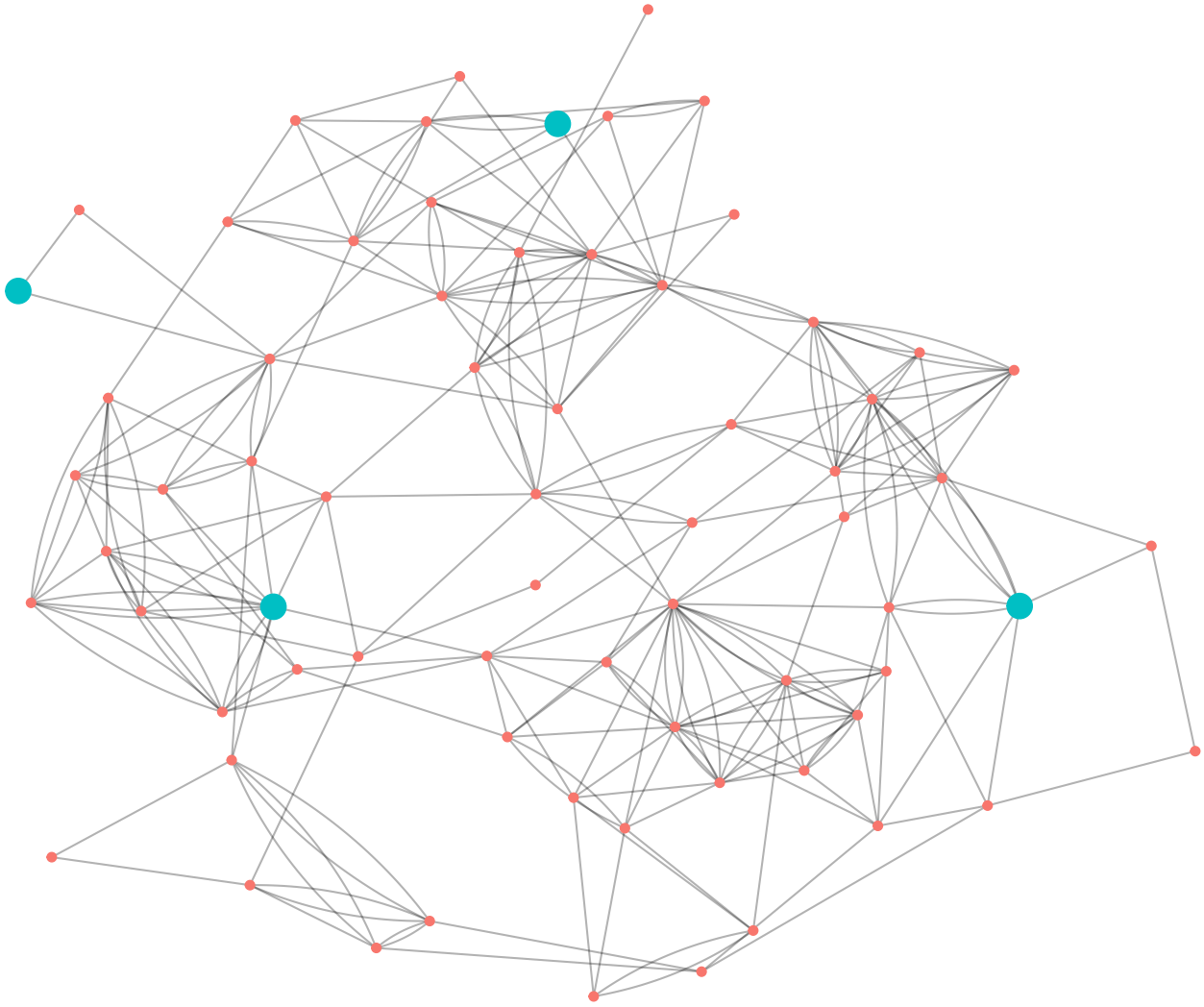
DPP samples



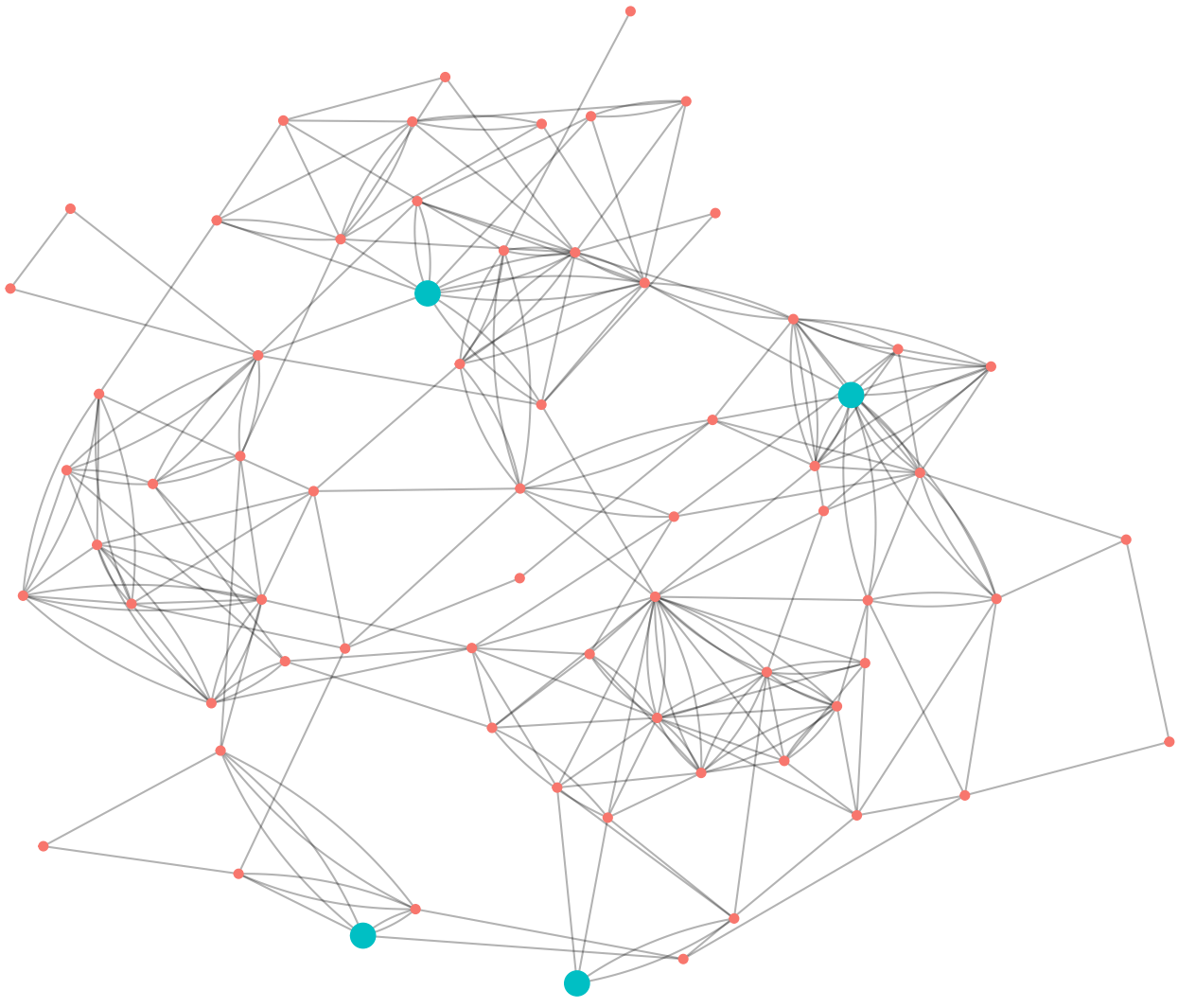
DPP samples



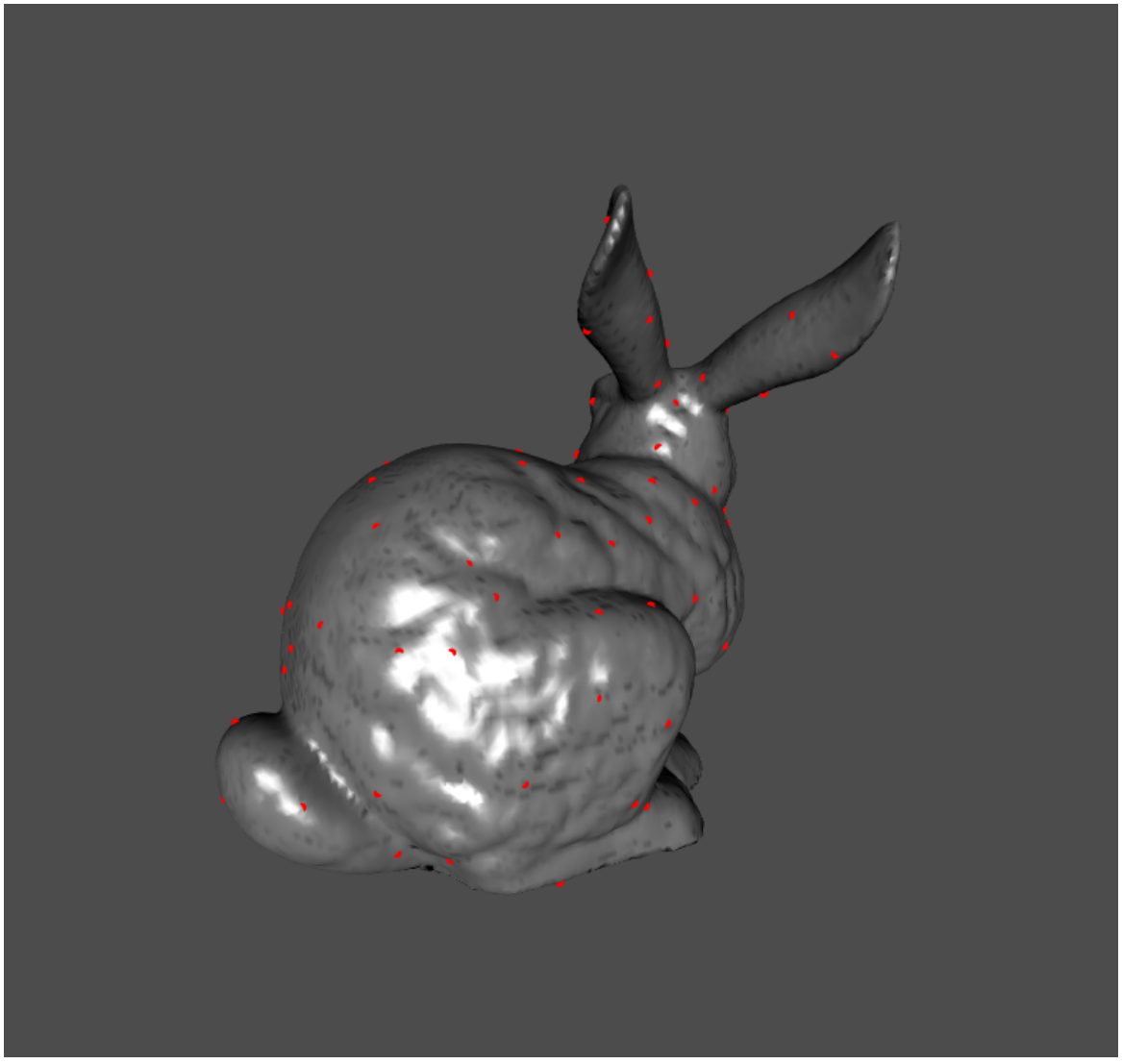
DPP samples



DPP samples

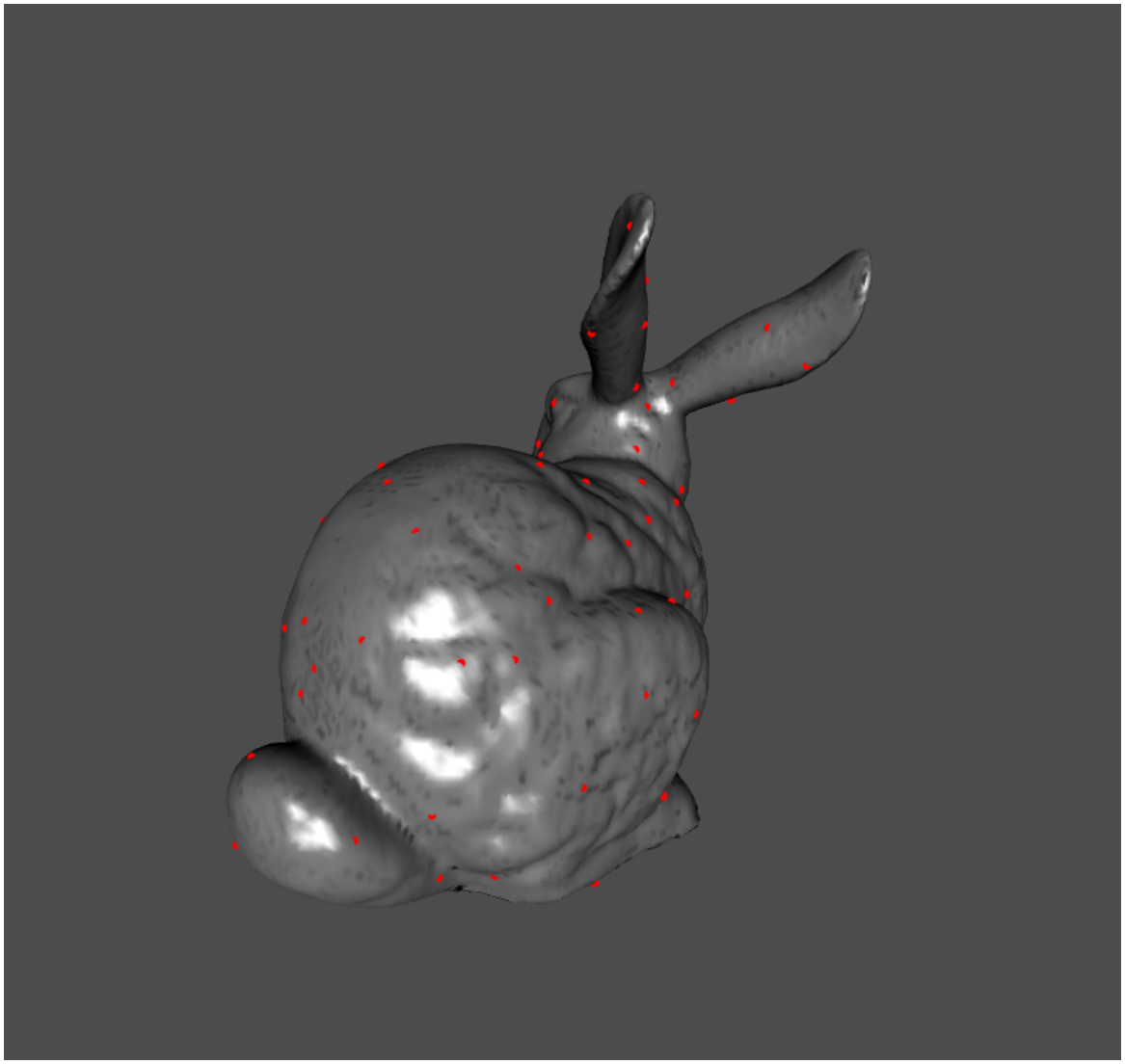


A large-scale example



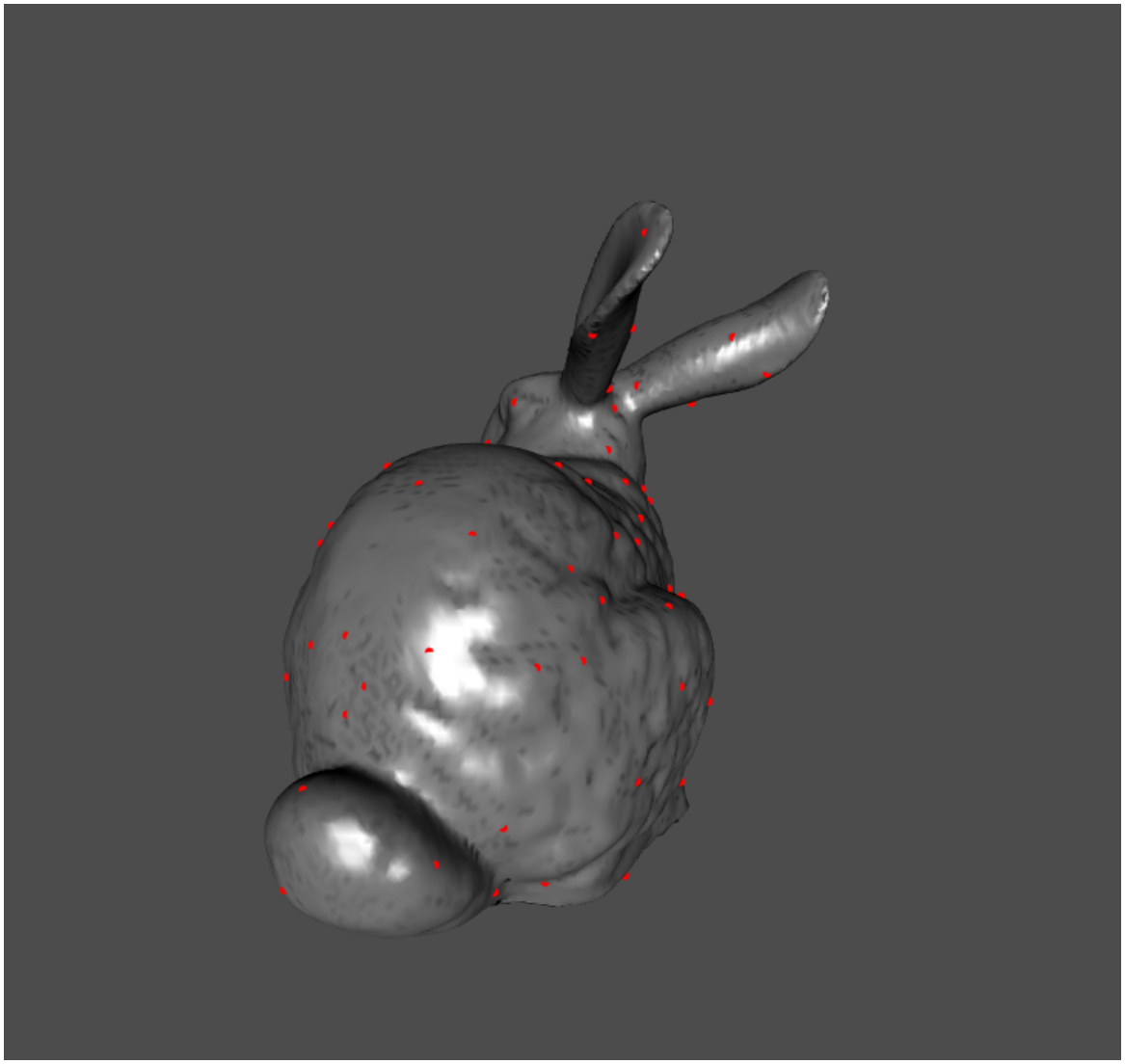
Stanford bunny (mesh with 30k vertices).

A large-scale example



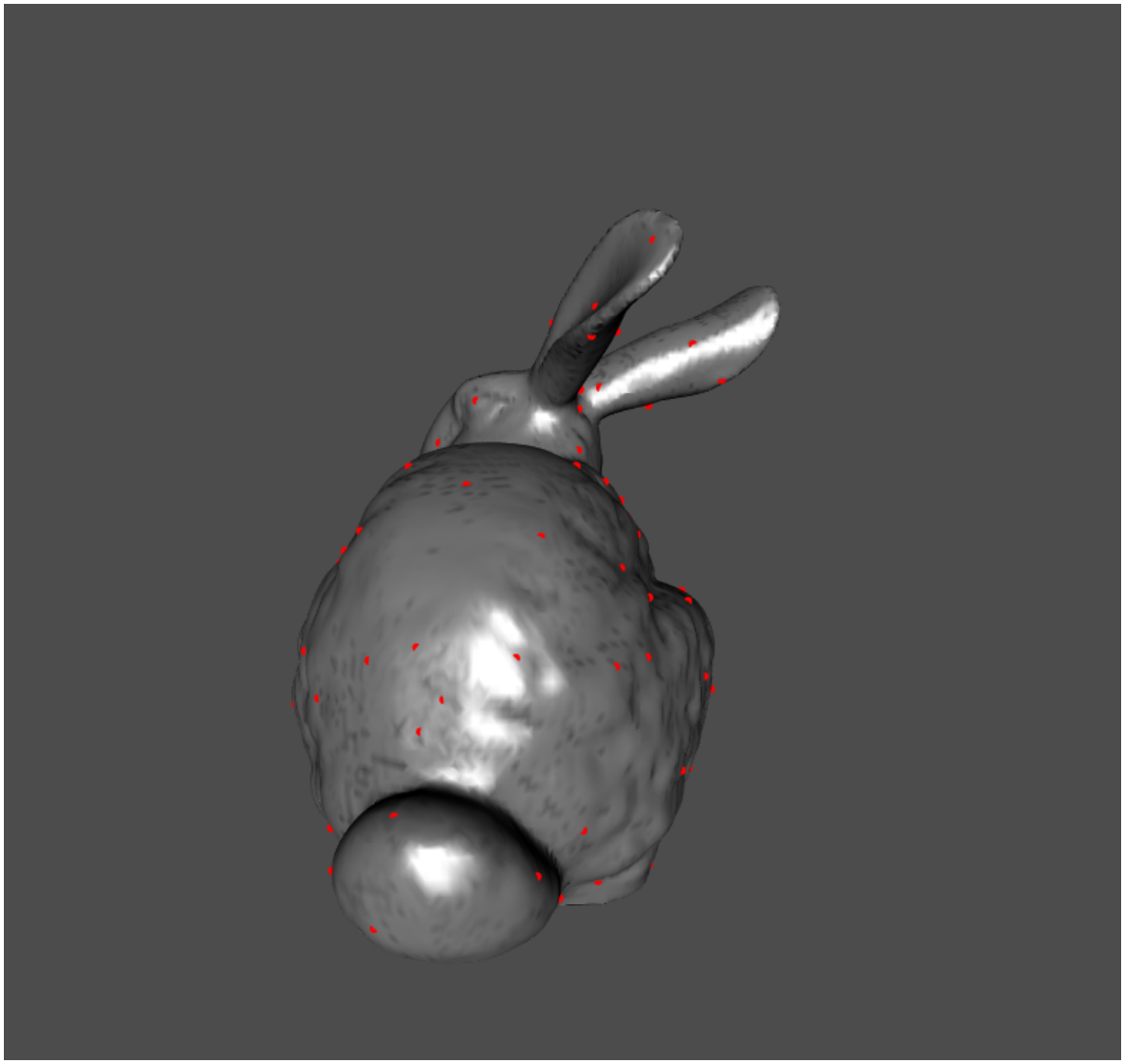
Stanford bunny (mesh with 30k vertices).

A large-scale example



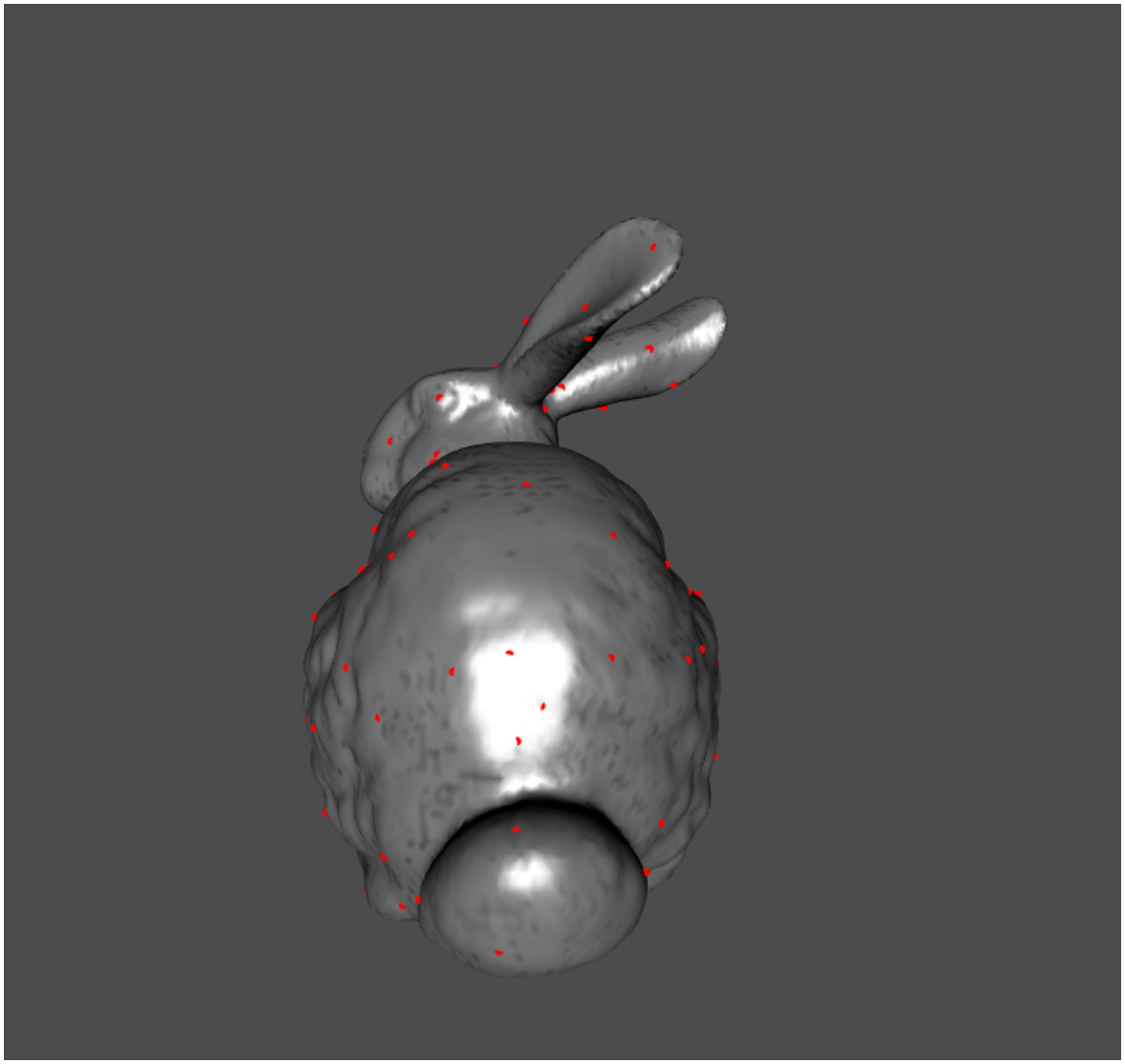
Stanford bunny (mesh with 30k vertices).

A large-scale example



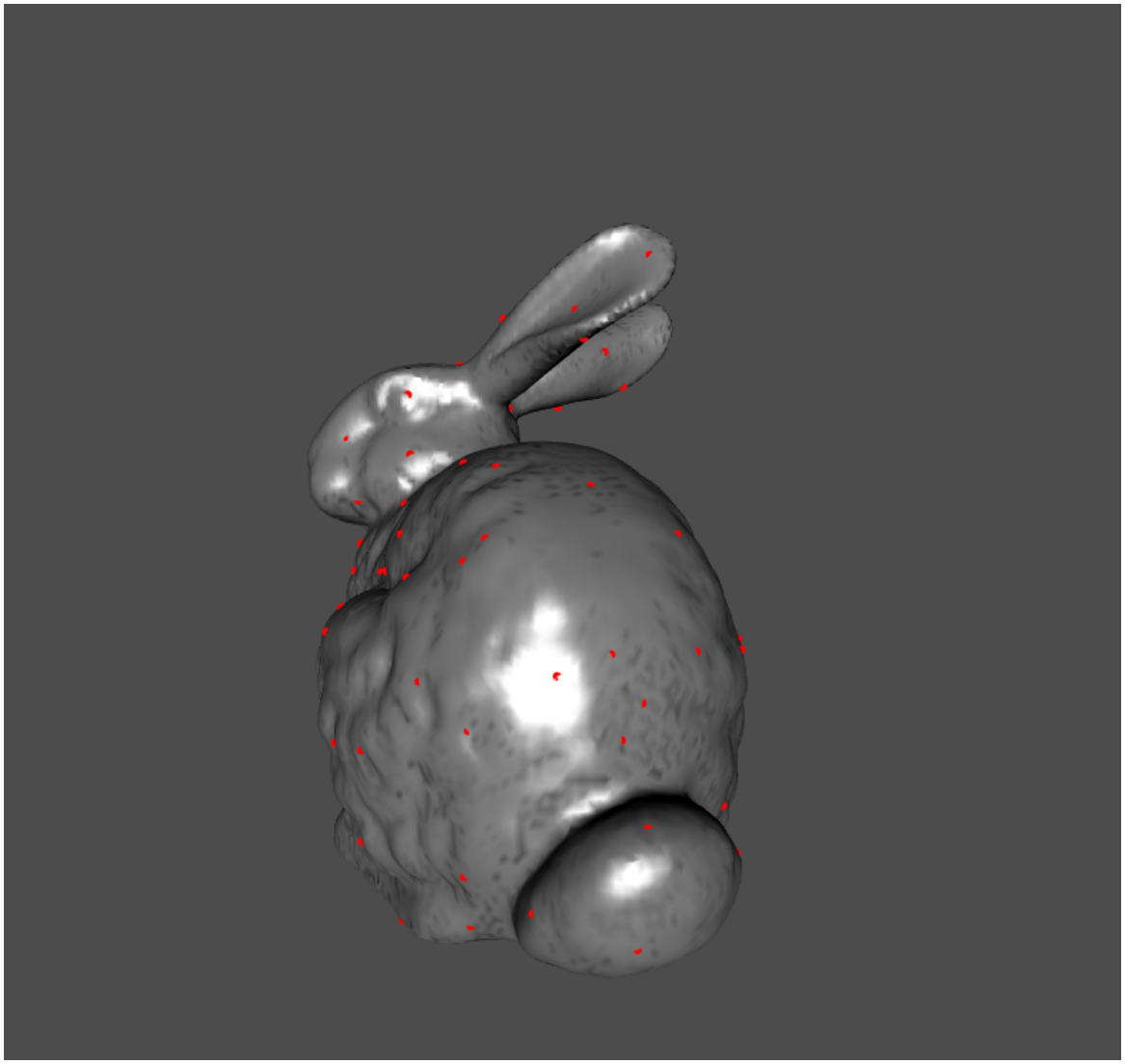
Stanford bunny (mesh with 30k vertices).

A large-scale example



Stanford bunny (mesh with 30k vertices).

A large-scale example



Stanford bunny (mesh with 30k vertices).

Practical considerations

- In practice the cost is often too high on large graphs, due to the eigendecomposition
- In Tremblay et al. (2017) we give an approximate algorithm that uses powers of the adjacency matrix
- See paper for details

Building coresets

- Here we are in a machine learning setting, and the goal is to find the minimum of a cost function:

$$C(\theta) = \sum_{i=1}^{n} f(x_i, \theta)$$

- i sums over data-points
- We seek argmin $C(\theta)$, but minimisation is expensive (because n is very large)

Building coresets

- "Coresets" retain a weighted subset of the datapoints, to create an approximate cost function:

$$\tilde{C}(\theta) = \sum_{j \in \mathcal{X}} w_j f(x_j, \theta)$$

- Minimisation is cheaper!
- Requirement: for all $\theta \in \Theta$

$$\left| \frac{\tilde{C}(\theta)}{C(\theta)} - 1 \right| \leq \epsilon$$

with high prob.

- Finding a procedure that outputs coresets is problem-specific!

Building coresets

- Many algorithms for building coresets proceed in this fashion (Munteanu and Schwiegelshohn, 2017):
  1. Do a first pass over data, computing a heuristic that gives each item a certain priority (optimal strategy is to give high priority to "unusual" - high leverage - items)
  2. Sample k items independently, with higher probability for high-priority items
  3. Set $w_i = p_i^{-1}$, i.e. importance sampling.
- Tremblay et al. (2018): simply replace independent samples with samples from a DPP
- Can prove that resulting random sets have the coreset property with high probability, theoretical arguments suggest that performance should improve (practical experiments show it does).

Challenges & questions

- I glossed over the fact that standard DPPs produce sets of *random*, not fixed size.
- Fixed size: so-called "k-DPPs" (Kulesza & Taskar, 2011) are less tractable (inclusion probabilities are harder).
- In recent work we show asymptotic equivalence of fixed-size and varying size DPPs (Barthelmé et al., 2018) which is good news
- However problems remain:
  - Hyperparameters (in the kernel function)
  - Speeding up sampling
  - Behaviour in high dimensions

Conclusion

- DPPs are an interesting class of tools for sampling
- As far as point processes go, they are fairly tractable
- They have fascinating links to Gaussian Processes, random matrix theory, graph theory, optimal design theory, stat. physics
- I've shown applications to graphs and coresets, but there are many more
- Some theoretical challenges remain

Post-doc position

- Post-doc position available in Grenoble, working with Ronald Phlypo & myself.
- Topic is Gaussian processes on graphs, come see me!

References

Barthelmé, S., Amblard, P.-O., and Tremblay, N. (2018). Asymptotic equivalence of fixed-size and varying-size determinantal point processes. *arXiv preprint arXiv:1803.01576*.

Kulesza, A. and Taskar, B. (2012). Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 5(2–3):123–286.

Li, C., Jegelka, S., and Sra, S. (2016). Efficient Sampling for k-Determinantal Point Processes.

Munteanu, A. and Schwiegelshohn, C. (2017). Coresets-Methods and History: A Theoreticians Design Pattern for Approximation and Streaming Algorithms. *KI - KÃŒnstliche Intelligenz*.

Tremblay, N., Amblard, P.-O., and Barthelme, S. (2017a). Graph sampling with determinantal processes. working paper or preprint.

Tremblay, N., Barthelme, S., and Amblard, P.-O. (2017b). Échantillonnage de signaux sur graphes via des processus déterminantaux. In *GRETSI*, Juan-les-Pins, France.