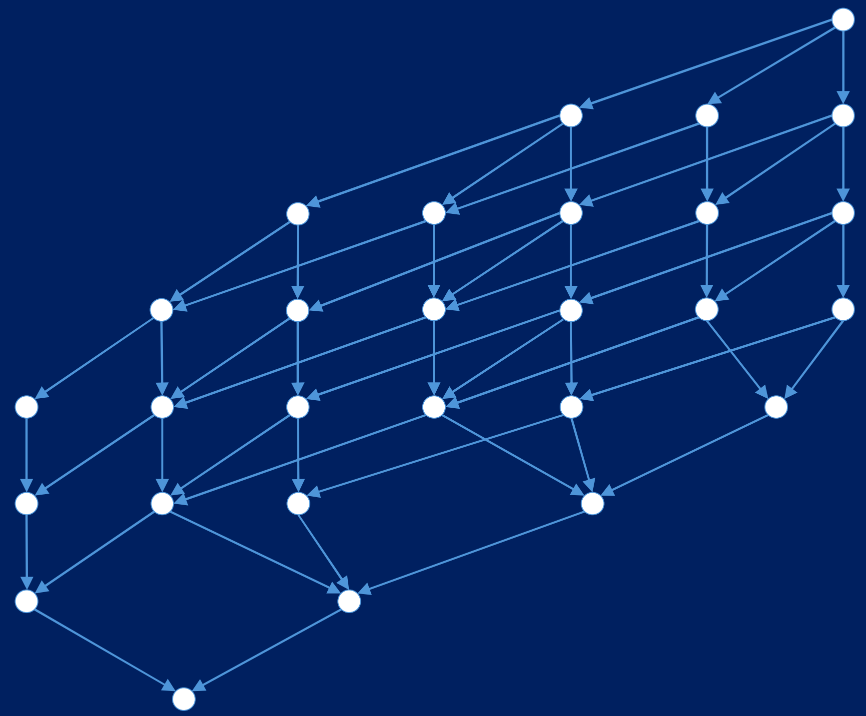
Graph Signal Processing

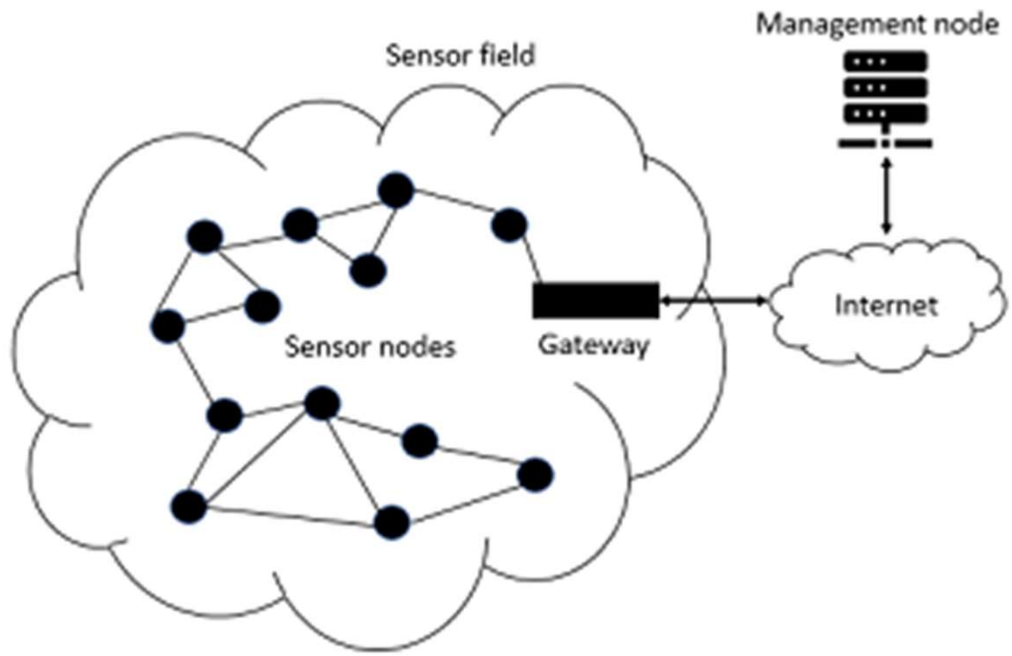
Bastian Seifert

Sommersemester 2025



Problem statement

- Sensors might be expensive to operate
- Sensor/battery lifetime might be increased when used less often
- System might be more efficient with less data transmitted



https://study.com/cimages/multimages/16/c701edbb-2f75-4518-a658-60a5743f66a5_sensor_networ.png

Sampling problem: How to infer values of nodes not observed?

Sampling

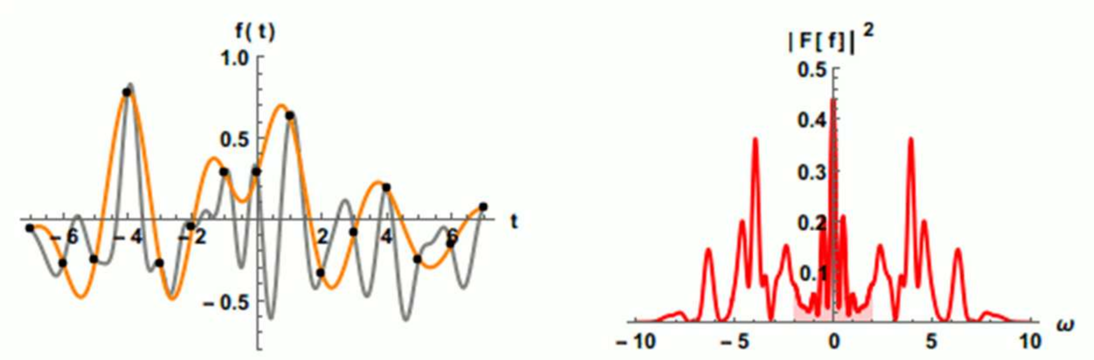
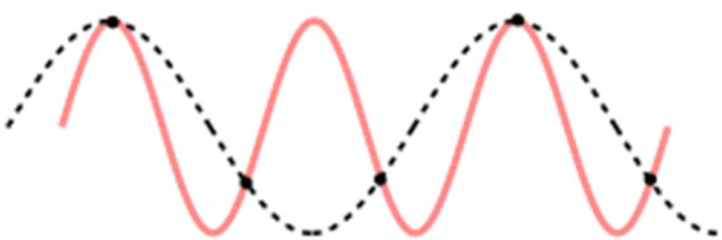
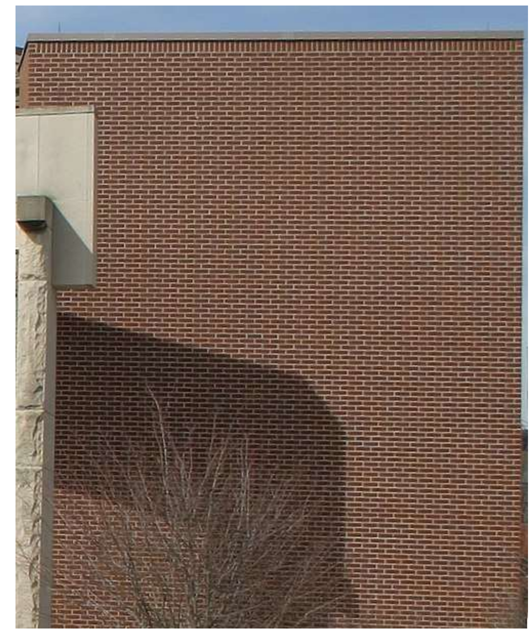


Figure: https://en.wikipedia.org/wiki/Nyquist%E2%80%93Shannon_sampling_theorem

Down-Sampling (gone wrong)





Figures: https://en.wikipedia.org/wiki/Nyquist%E2%80%93Shannon_sampling_theorem

Sampling

$x \in \mathbb{C}^N$

$$\mathbf{x}_{\mathcal{M}} = \Psi \mathbf{x} \in \mathbb{C}^M \quad , M << N$$

↑ Measured signal

Sampling takes (a linear combination of) samples at choosen measurement points to produce a lower dimensional signal.

Interpolation

$$\tilde{\mathbf{x}} = \Phi \mathbf{x}_{\mathcal{M}} = \Phi \Psi \mathbf{x} \in \mathbb{C}^N$$

Interpolation approximates the original signal based on the sampled signal.

$\tilde{x} \approx x$

Example

$$\Psi = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \qquad \Psi \mathbf{x} = \begin{bmatrix} x_2 + x_4 \\ x_1 + x_3 \end{bmatrix} \qquad \Phi = \begin{bmatrix} 0 & \frac{1}{2} \\ \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \\ \frac{1}{2} & 0 \end{bmatrix} \qquad \Phi \Psi \mathbf{x} = \frac{1}{2} \begin{bmatrix} x_1 + x_3 \\ x_2 + x_4 \\ x_1 + x_3 \\ x_2 + x_4 \end{bmatrix}$$

$$x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}$$

Perfekte Rekonstruktion für Signale mit $x_1 = x_3$ und $x_2 = x_4$

Sampling and Interpolation are pseudo-inverse operations to each other

Bandlimited Graph Signals

Graph signals are called k-bandlimited if they can be expressed as a linear combination of k eigenvectors of the graph shift operator

k Eigenvektoren von T, also graph Frequenzen

$$\mathbf{x} = U_k \alpha, \quad \text{with } \alpha \in \mathbb{R}^k$$

Die k Frequenzen können beliebig gewählt werden

Note that this is equivalent to k-sparsity in the graph Fourier domain

$$||\mathcal{F}\mathbf{x}||_0 = k$$

(Hint: Sparse Fourier transformation)

Frequenzbereich der $\neq 0$ ist definiert

Perfect Reconstruction Condition

k-bandlimited

A graph signal can be perfectly recovered if

$$\text{rank}(\Psi U_k) = k$$

The interpolation operator which achieves perfect recovery is

$$\Phi = U_k V$$

where

$$V \Psi U_k = \mathbb{1}_k$$

is an identity matrix (i.e., a pseudo-inverse)

$$\Phi \Psi x = \Phi \Psi U_k \alpha = U_k \underbrace{V \Psi U_k}_{\mathbb{1}} \alpha = U_k \alpha = x$$

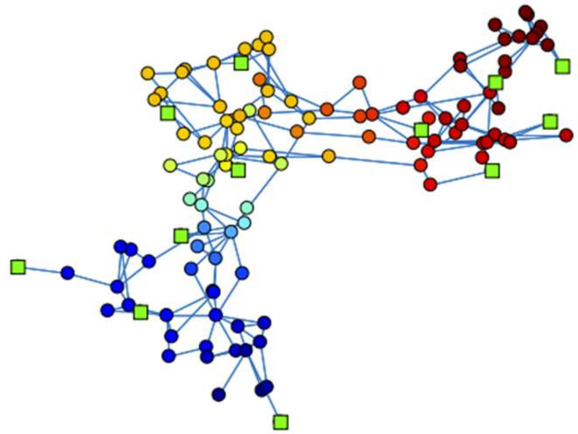
Perfect reconstruction is possible for bandlimited signals

Sampling strategies

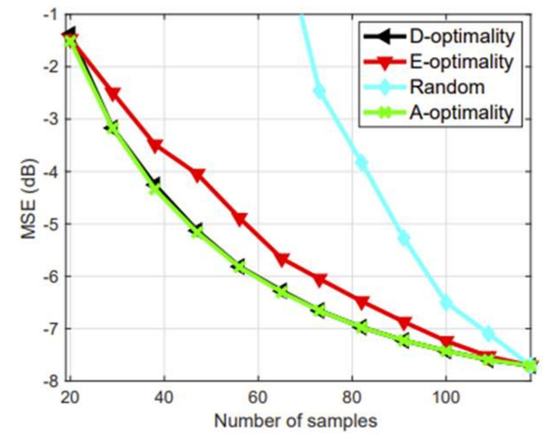
- **D-optimal design**: Minimizes the log-determinant of the interpolation error covariance to achieve maximum overall confidence in parameter estimation
- **A-optimal design**: Minimizes the mean squared error across all estimated coefficients to give the best average estimation accuracy
- **E-optimal design**:Minimize the maximum (worst-case) variance in any direction, ensuring robust reconstruction under the least favorable scenario

Algorithm 1 : Greedy selection of graph samples

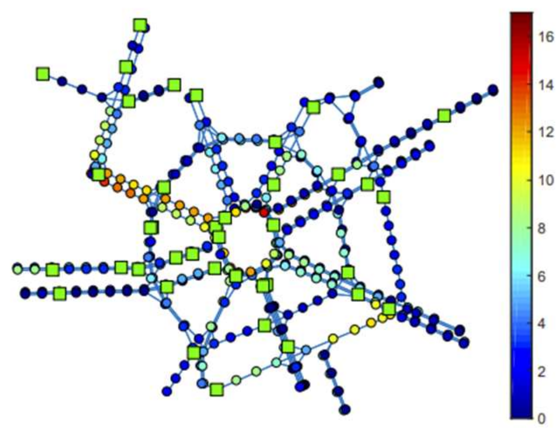
Input Data : $\mathbf{U}_{\mathcal{F}}$, M;
Output Data : $\mathcal{S}$, the sampling set.
Function : initialize $\mathcal{S} \equiv \emptyset$
while $|\mathcal{S}| < M$
$s = \arg\max_{j} f(\mathcal{S} \cup \{j\})$;
$\mathcal{S} \leftarrow \mathcal{S} \cup \{s\}$;
end



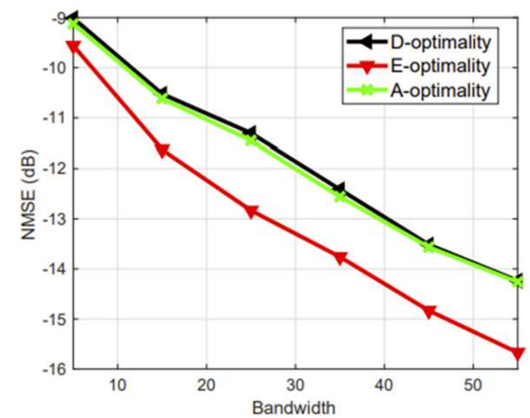
(a) Graph topology and sampling set



(b) MSE versus number of samples



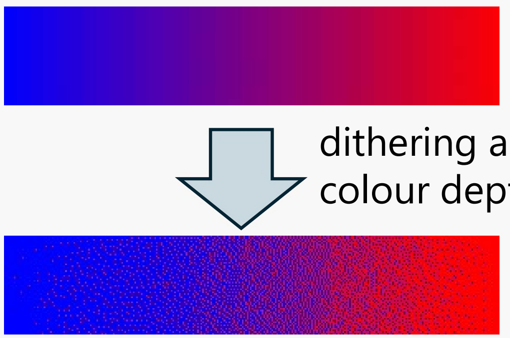
(a) Graph topology and sampling set



(b) NMSE versus bandwidth

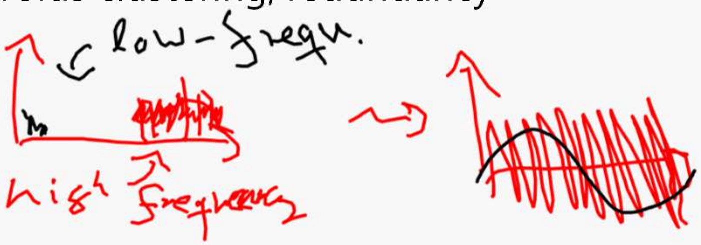
Figures: Sampling and Recovery of Graph Signals, Lorenzo et al.

Dithering and Blue noise



dithering approximates higher colour depth

•Human perception prefers **even but non-uniform** sample distributions
•Blue noise = no low-frequency components → avoids clustering, redundancy



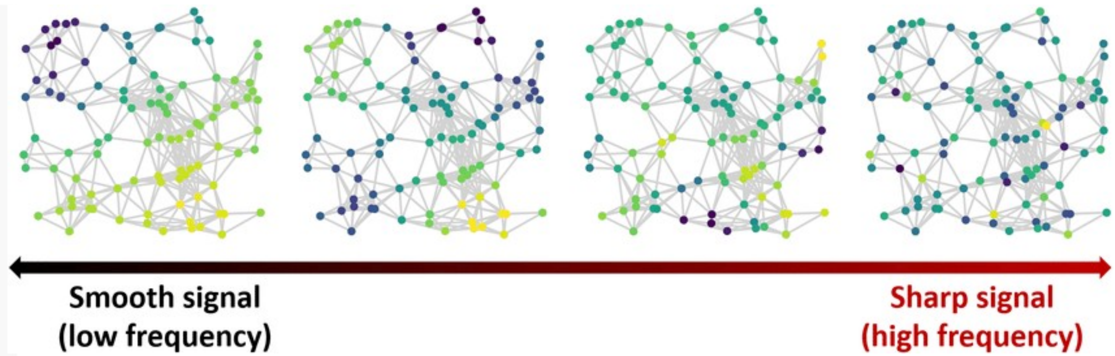


Figure: https://www.researchgate.net/publication/355901199/figure/fig1/AS:1086260500275200@1635996093701
graph-signal-becomes-sharper-going-from-left-to-right-Smooth-graph-signals-assign.ppm

Blue noise sampling

•Blue noise sampling patterns have energy concentrated in high frequencies
• Select nodes so that the frequency spectrum of the sampling pattern avoids low frequencies, preserving signal diversity and reducing aliasing

Blue noise sampling spreads samples to avoid low-frequency redundancy and clustering

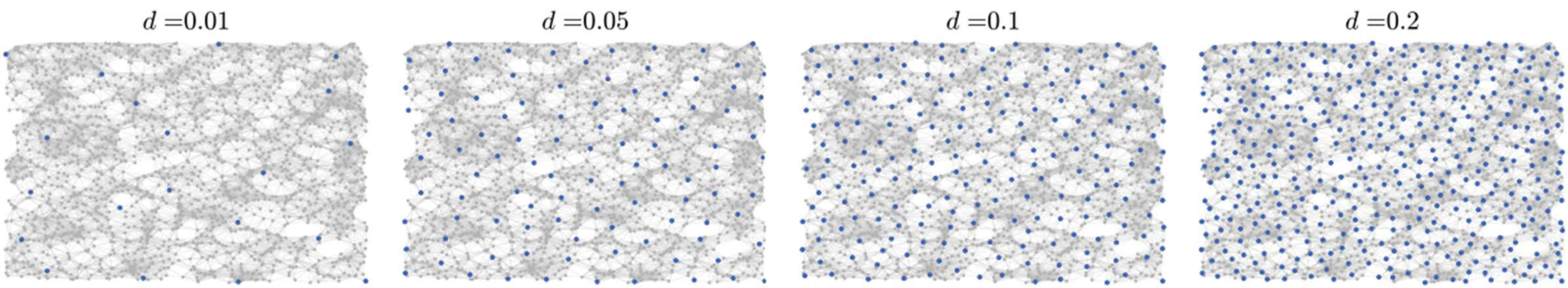
Energy Penalization Method

Blue-noise sampling can be framed as minimizing overlap with low-frequency eigenspaces:

$$\min_{s\in\{0,1\}^n} \|U_{low}^\top s\|^2$$

Spectral Spreading Principle

- Maximize energy outside the low-frequency subspace by choosing sampling sets that "look like noise" in the spectral domain
- Ensure samples are well-distributed across the graph topology



Figures: Blue-Noise Sampling on Graphs, Parada-Mayorga et al.

Blue-noise sampling penalizes low-frequency concentration in sampling vector.

Algorithm

Idea: Sequential computation of distances between sampling points and relocation of points with short geodesic distance to put them far from each other.

Algorithm 1: Void and Cluster Algorithm for Graphs.

Input: m: number of samples, σ, `NumIter`.

Output: s: sampling pattern

Initialisation : $\boldsymbol{s} = \mathbf{0}$, `IndA` $= -1$, `IndB` $= -1$.

Calculate $\mathbf{K}(i,j) = e^{-\frac{\Gamma(i,j)^2}{\sigma}}$ for all $1 \leq i, j \leq N$.

$\mathbf{c} = \mathbf{K}\mathbf{1}_{N\times 1}$.

Get $\mathcal{M}$ as m nodes selected at random.

$\boldsymbol{s}(\mathcal{M}) = 1$. Indiziert ob Element in sampling set

for $r = 1:1:$ `NumIter` **do**

$\mathbf{c}(\text{supp}(\boldsymbol{s})) = \sum \mathbf{K}(\text{supp}(\boldsymbol{s}), \text{supp}(\boldsymbol{s}))$. Berechnet, wie nah Sampling Punkte liegen

$\mathbf{c}(\text{supp}(\boldsymbol{s})^c) = \sum \mathbf{K}(\text{supp}(\boldsymbol{s}), \text{supp}(\boldsymbol{s})^c) - \tau$. wie nah Punkte außerhalb sampling set sind

$\boldsymbol{s}(\arg\max_i\{\mathbf{c}(i)\}) = 0$. schlechtester sampling point

$\boldsymbol{s}(\arg\min_i\{\mathbf{c}(i)\}) = 1$. bester Kandidat für sampling point

if `IndA` $= \arg\max_i\{\mathbf{c}(i)\}$ and `IndB` $= \arg\min_i\{\mathbf{c}(i)\}$ **then**

break

else

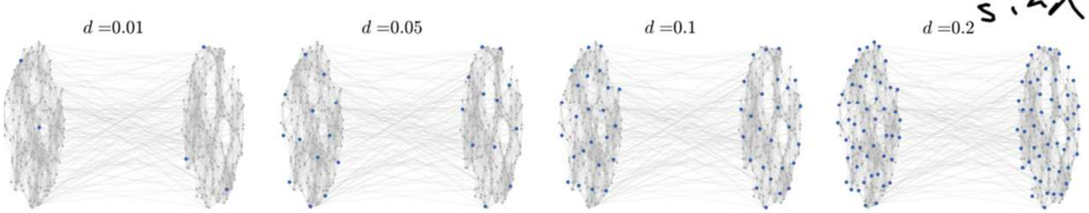
`IndA`$= \arg\min_i\{\mathbf{c}(i)\}$.

`IndB`$= \arg\max_i\{\mathbf{c}(i)\}$.

end if

end for

return s



Figures: Blue-Noise Sampling on Graphs, Parada-Mayorga et al.

Algorithm is computationally more efficient than greedy sampling