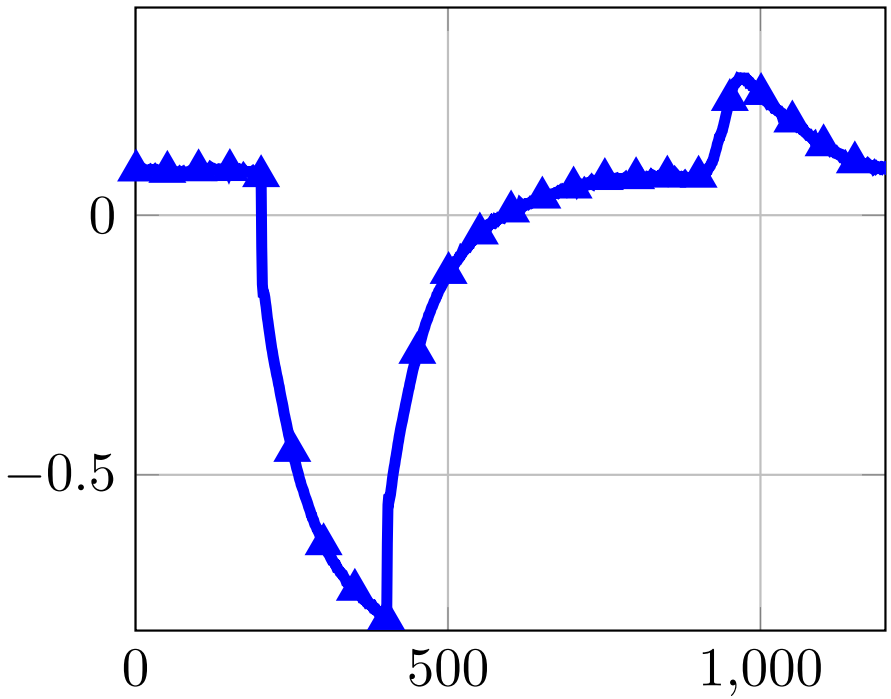
Reduction of Necessary Data Rate for Neural Data through Exponential and Sinusoidal Spline Decomposition using the Finite Rate of Innovation Framework

T. Schnier C. Bockelmann A. Dekorsy

Department of Communications Engineering
University of Bremen

New Orleans, March 7th, 2017

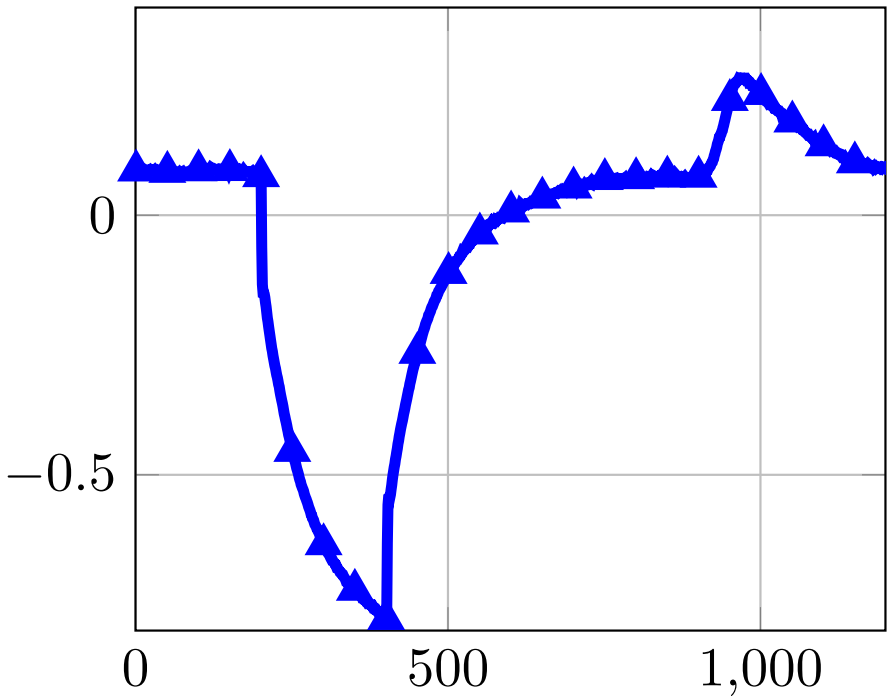
Motivation: Action potential



Observations

- State of the Art (wireless): Only Spike Detection possible
- Strict hardware requirements: Fine reconstruction impossible
- Compressed Sensing (CS): Reduce data rate by factor $3 - 5$

Motivation: Action potential



Observations

- State of the Art (wireless): Only Spike Detection possible
- Strict hardware requirements: Fine reconstruction impossible
- Compressed Sensing (CS): Reduce data rate by factor $3 - 5$

Main Contribution

Finite Rate of Innovation (FRI): Data rate reduction by factor **200**

Table of Contents

Finite Rate of Innovation

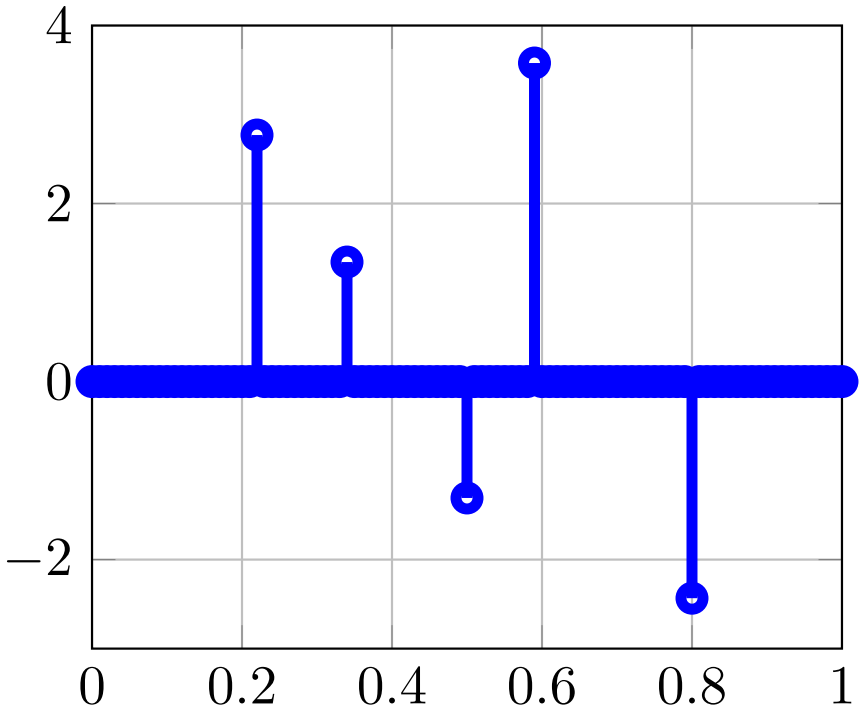


Figure: Stream of Diracs

Formal definition

- Set of *known* functions: $\{\mathbf{\Psi}_r(t)\}_{r=0,\dots,R-1}$

$$\mathbf{x}(t) = \sum_{i\in\mathbb{Z}} \sum_{r=0}^{R-1} c_{ir}\, \mathbf{\Psi}_r\left(\frac{t-t_i}{T}\right)$$

- Rate of Innovation ρ: Number of unknowns
- Spike train: Finite $\rho = 10$, unlimited bandwidth

Exponential and sinusoidal splines

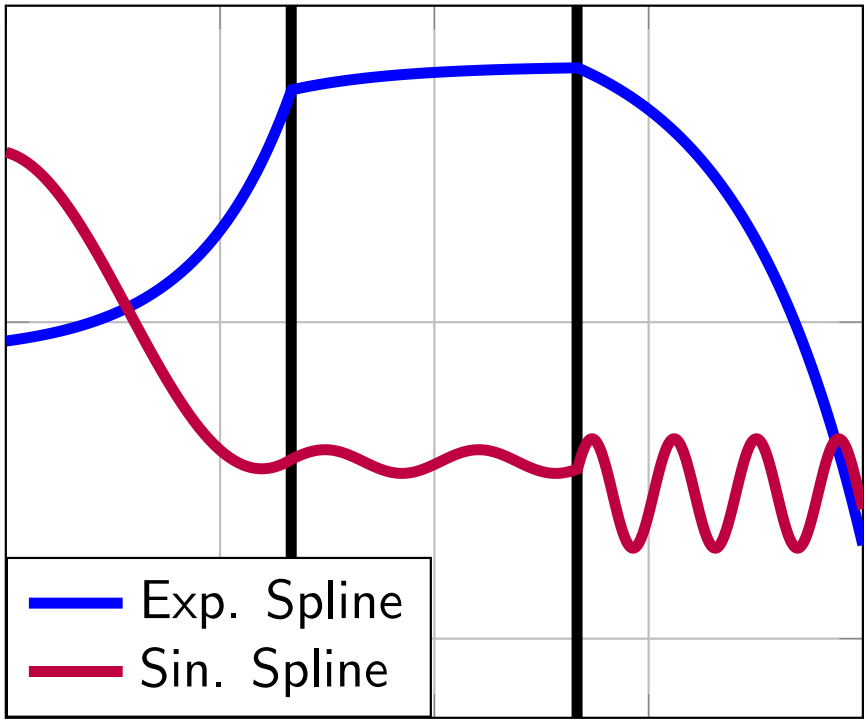


Figure: Example splines

Exponential and sinusoidal splines

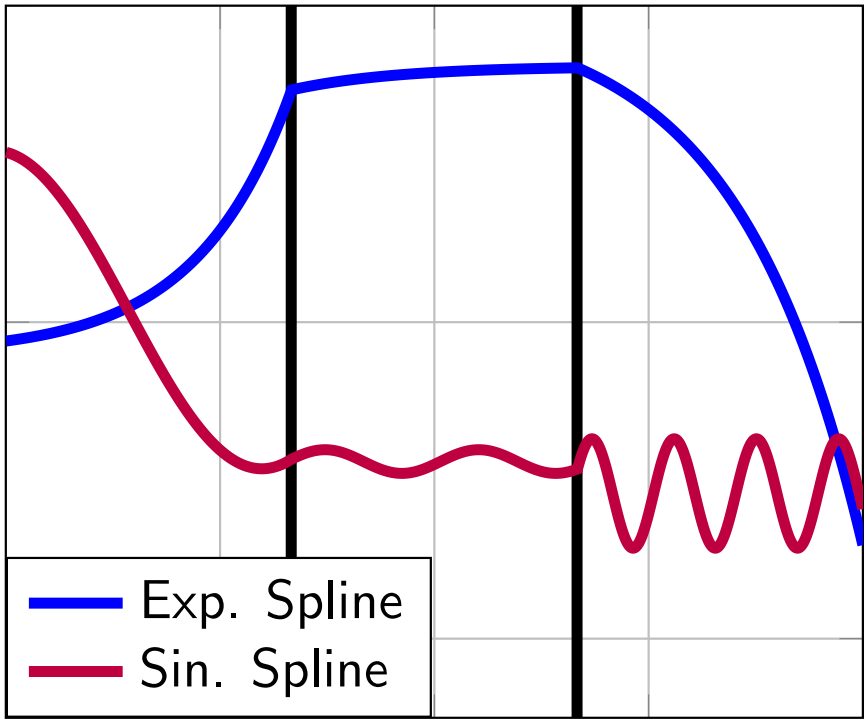


Figure: Example splines

Formal definition

Exponential splines:

$$\Psi_i(t) = a_i e^{b_i t} + d_i$$

for $t_i \leq t < t_{i+1}$

Sinusoidal splines:

$$\Psi_i(t) = a_i \cos(b_i t + c_i) + d_i$$

for $t_i \leq t < t_{i+1}$

Continuous at knots:

$$\mathbf{x}_i(t_i) = \mathbf{x}_{i+1}(t_i)$$

Needed steps

1. Find nonlinear parts b_i
2. Solve linear regression for remaining unknowns
3. Find knots of the spline

Reconstruction of single part

Problem formulation

Given:

$$\mathbf{x}(t) = \sum_{i=1}^{k} a_i e^{b_i t}$$

and $y_i = \langle \mathbf{x}(t), \mathbf{h}(t + iT) \rangle + n$ for $1 \leq j \leq 2k$.

Task: Find **unknown** a_i and b_i

Idea: Build annihilating filter

$$(\mathbf{g}(t) * \mathbf{x}(t))[jT] = 0 \, \forall 1 \leq j \leq k$$

Then: Roots of $\mathbf{g}(t)$ are e^{b_i}!

Reconstruction of single part

Problem formulation

Given:

$$\mathbf{x}(t) = \sum_{i=1}^{k} a_i e^{b_i t}$$

and $y_i = \langle \mathbf{x}(t), \mathbf{h}(t + iT) \rangle + n$ for $1 \leq j \leq 2k$.

Task: Find **unknown** a_i and b_i

Idea: Build annihilating filter

$$(\mathbf{g}(t) * \mathbf{x}(t))[jT] = 0 \, \forall 1 \leq j \leq k$$

Then: Roots of $\mathbf{g}(t)$ are e^{b_i}!

Solution: Annihilating Filter

Rewrite convolution to

$$\begin{pmatrix} \mathbf{y}_1 & \cdots & \mathbf{y}_k \\ \vdots & \ddots & \vdots \\ \mathbf{y}_k & \cdots & \mathbf{y}_{2k-1} \end{pmatrix} \begin{pmatrix} g_{k-1} \\ \vdots \\ g_0 \end{pmatrix} = \begin{pmatrix} \mathbf{y}_{k+1} \\ \vdots \\ \mathbf{y}_{2k} \end{pmatrix}$$

- Solve for $\mathbf{g}(t)$, find roots to get e^{b_i}
- With known b_i, search for a_i becomes **linear**
- Works for every filter $\mathbf{h}(t)$!

Exponential functions

Problem formulation

Unknown coefficients:

$$\begin{aligned}\mathbf{x}(t) &= ae^{bt} + d \\ &= a(e^b)^t + d(e^0)^t\end{aligned}$$

FRI: $\rho = 3$
Given for $i = 1, 2, 3$:

$$y_i = \langle \mathbf{x}(t), \mathbf{h}(t + iT) \rangle + n$$

Annihilating Filter

Build the annihilating filter

$$\begin{aligned}\mathbf{g}(t) &= (t - e^b)(t - e^0) \\ &= t^2 + t(-e^b - 1) + e^b\end{aligned}$$

Find roots with

$$\begin{pmatrix} y_3 & y_2 & y_1 \end{pmatrix} \begin{pmatrix} 1 \\ -e^b - 1 \\ e^b \end{pmatrix} = 0$$

Solution

$$b = \log\left(\frac{y_3 - y_2}{y_2 - y_1}\right) / T$$

Sinusoidal functions

Problem formulation

Unknown coefficients:

$$\begin{aligned}\mathbf{x}(t) &= a\cos(bt+c)+d \\ &= \frac{a}{2}e^{ic}(e^{ib})^t + \frac{a}{2}e^{-ic}(e^{-ib})^t \\ &\quad + d(e^0)^t\end{aligned}$$

FRI: $\rho = 4$

Given for $i = 1, 2, 3, 4$:

$$y_i = \langle \mathbf{x}(t), \mathbf{h}(t+iT)\rangle + n$$

Annihilating Filter

Define $\hat{b} := 1 + 2\cos(b)$

$$\begin{aligned}\mathbf{g}(t) &= (t-e^{ib})(t-e^{-ib})(t-e^0) \\ &= t^3 - \hat{b}t^2 + \hat{b}t - 1\end{aligned}$$

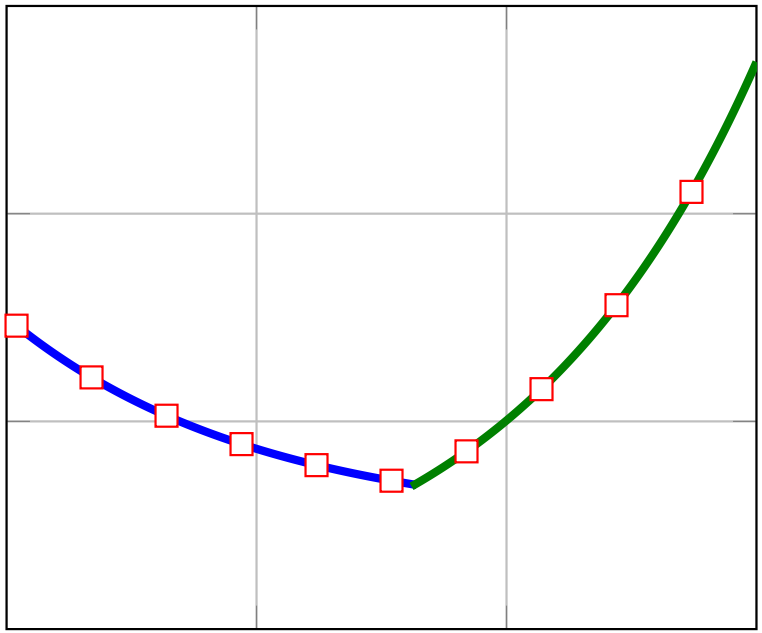
Find roots with

$$\begin{pmatrix} y_4 & y_3 & y_2 & y_1 \end{pmatrix} \begin{pmatrix} 1 \\ -\hat{b} \\ \hat{b} \\ -1 \end{pmatrix} = 0$$

Solution

$$b = \operatorname{acos}\left(\frac{1}{2}\left(\frac{y_4 - y_1}{y_3 - y_2} - 1\right)\right)/T$$

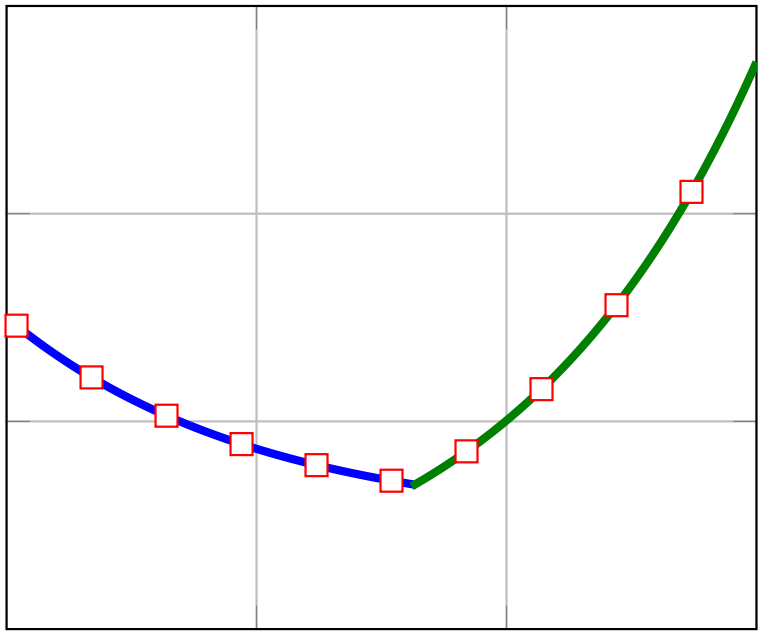
Finding the knots



Main idea

1. Compute parts for 4 samples
2. Find first sample not on function
3. Use Newton method to find intersection
4. Repeat

Finding the knots



Main idea

1. Compute parts for 4 samples
2. Find first sample not on function
3. Use Newton method to find intersection
4. Repeat

Sampling filter choice

Open question: Best sampling filter?

Sampling Filter

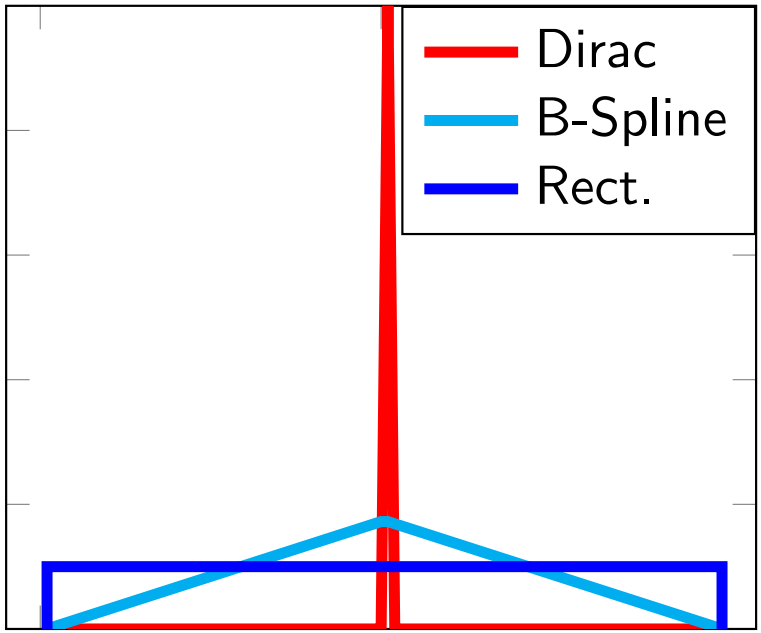


Figure: Different filter designs

Filter Design

- Algorithm works with any filter $\mathbf{h}(t)$
- Noise tolerance still dependents on filter
- What is the best filter to maximize the SNR?

Theoretical results

Matched Filter Design

- Theoretical best filter is matched filter

$$\mathbf{h}(t) = K\mathbf{x}^*(iT - t)$$

- Problem: $\mathbf{x}(t)$ is **unknown**
- Idea: Stochastic approach, match to **mean** function

Theoretical results

Matched Filter Design

- Theoretical best filter is matched filter

$$\mathbf{h}(t) = K\mathbf{x}^*(iT - t)$$

- Problem: $\mathbf{x}(t)$ is **unknown**
- Idea: Stochastic approach, match to **mean** function

Mean Filter Design

- Mean exponential function

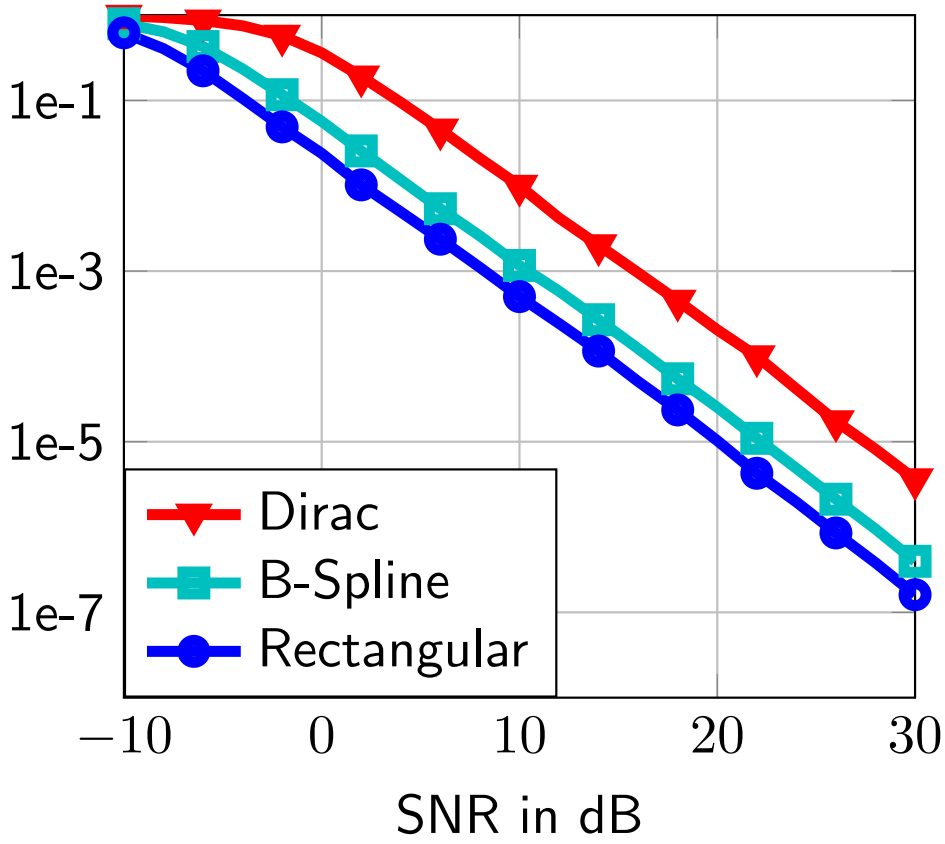
$$\mathbf{x}(t) = ae^{0t} + d = \text{const}$$

- Mean sinusoidal function

$$\mathbf{x}(t) = a\cos(0t + c) + d = \text{const}$$

- Solution: Best filter is **rectangular**!

Reconstruction for different filter designs



Simulation setup

- Filter width: $w = 100$
- Sample size: $m = 10$

Figure: Rec. error $||f - f_{\text{rec}}||_2$ vs. SNR

Reconstruction for different filter designs

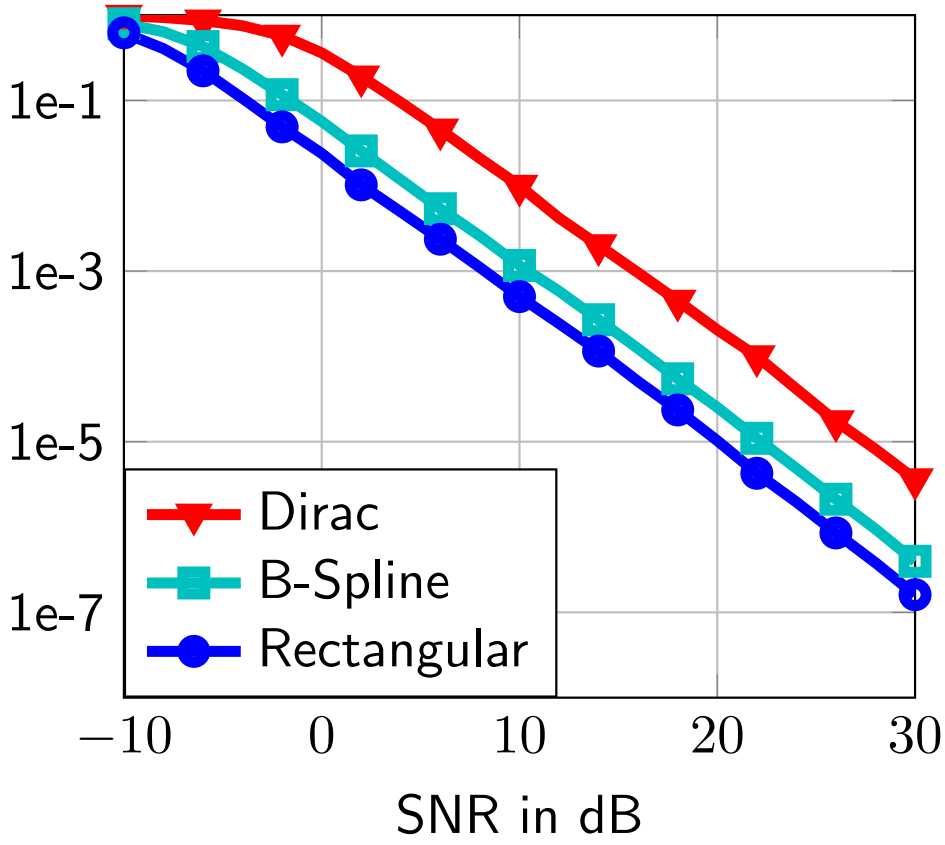


Figure: Rec. error $||f - f_{\text{rec}}||_2$ vs. SNR

Simulation setup

- Filter width: $w = 100$
- Sample size: $m = 10$

Observations

- Simulation confirms theoretical results
- Dirac is worst filter (no noise suppresion)
- Rectangular filter matches signal well

Reconstruction of neural data with a rect. filter

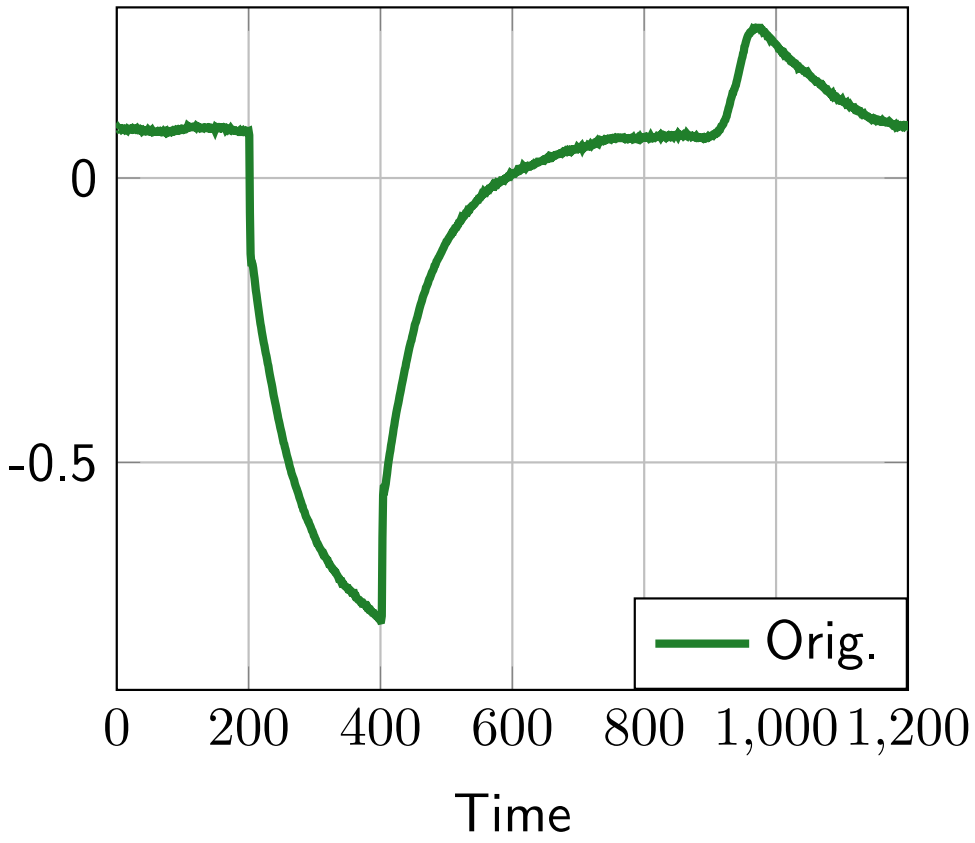


Figure: Reconstruction of neural data

Reconstruction of neural data with a rect. filter

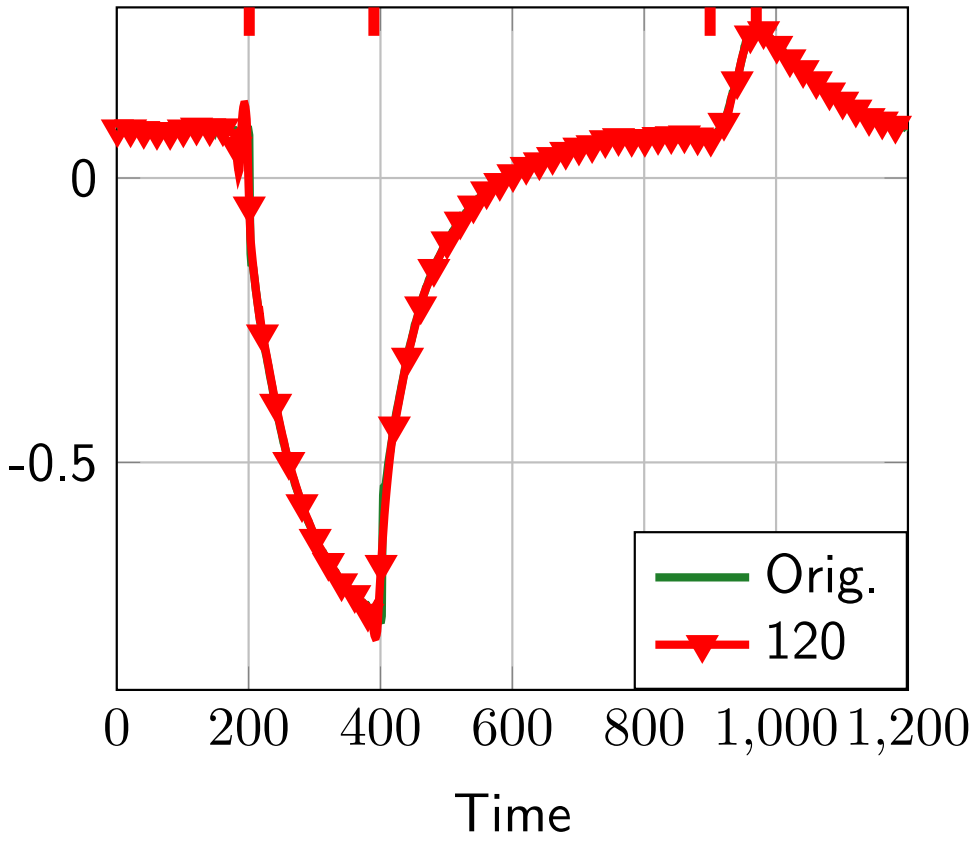


Figure: Reconstruction of neural data

Reconstruction of neural data with a rect. filter

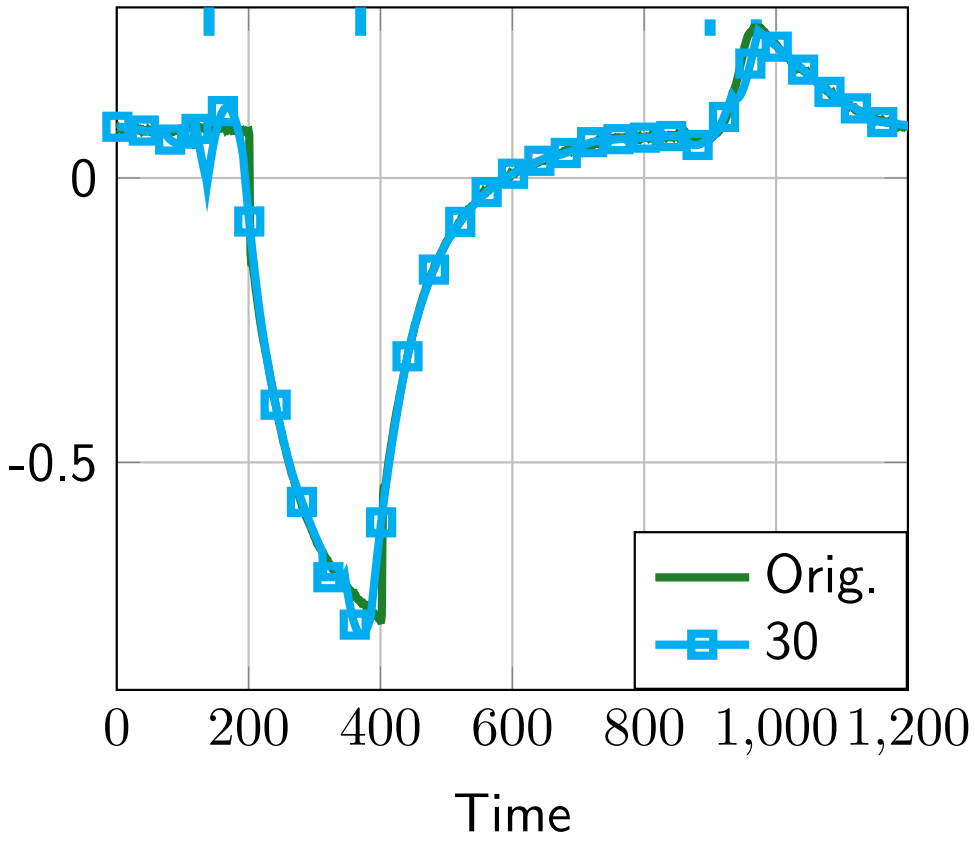
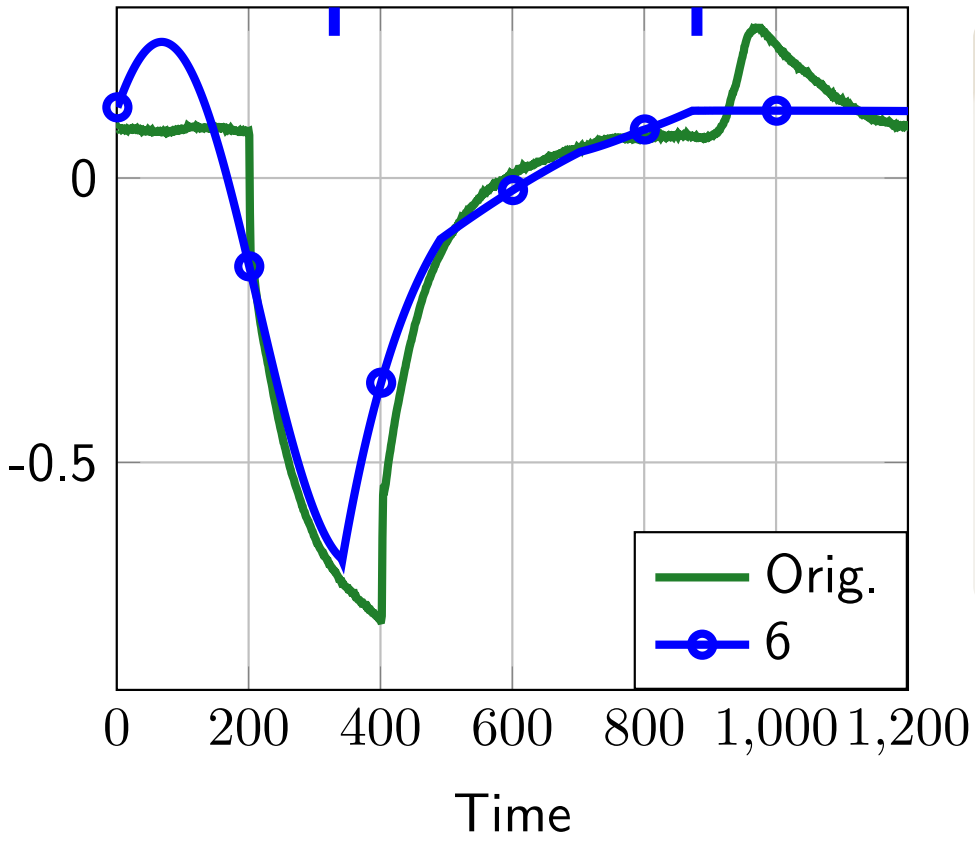


Figure: Reconstruction of neural data

Reconstruction of neural data with a rect. filter



Conclusion

- Reconstruction of neural data is possible!
- Rate reduction by factor 10 nearly without loss
- Factor 200 reduction still allows main feature extraction

Figure: Reconstruction of neural data