UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Optimizing Short-Time Fourier Transform Parameters via Gradient Descent

International Conference on Acoustics, Speech, & Signal Processing
Toronto, Ontario, 2021

An Zhao
Krishna Subramani
Paris Smaragdis

The problem

- Short Time Fourier Transforms need manual tuning
  - Size parameters (length and hop) are not differentiable

Discrete!!

$$F_W[m,k] = \sum_{n=-N/2}^{N/2} x[m+n]W_m[n]e^{-j\frac{2\pi}{N}kn}$$

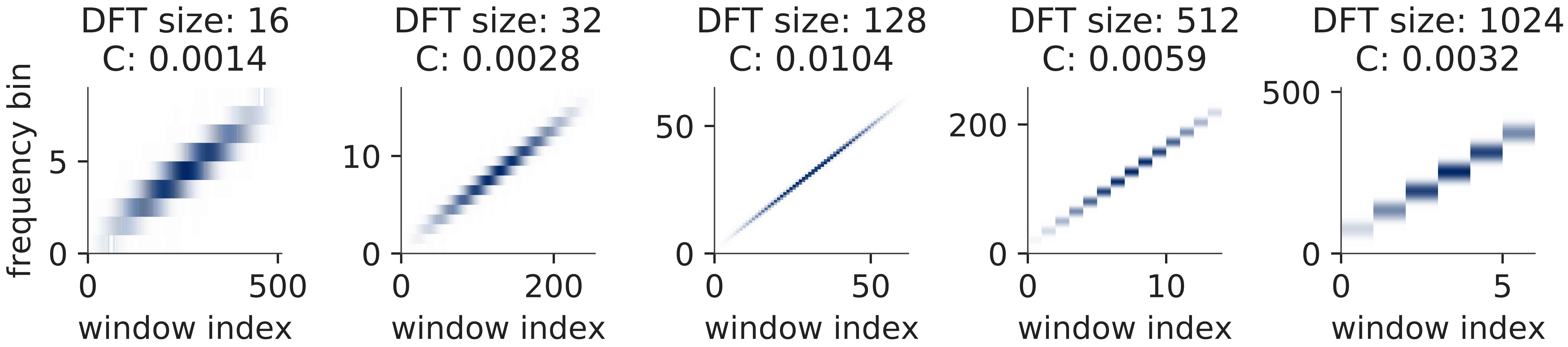
- But choosing proper values is crucial for many tasks

Our Goal

- Differentiable Short Time Fourier Transform sizes
  - Optimize window size N and hop size H
- We want to use gradient descent
  - Which will help incorporate this into deep network designs
    - e.g. auto-tune the STFT front-end of a denoiser or sound classifier
- But directly differentiating integer values is a no-go

A motivating problem

- Obtaining a sparse STFT representation



- Poor values for DFT size create a stepping effect; good value conveys the input signal better

Approaching the problem

- We use time/frequency sparsity as the cost function
  - We specifically use the kurtosis of the STFT output

$$C[m,W] = \frac{\sum_{k=0}^{N} |F_W[m,k]|^4}{(\sum_{k=0}^{N} |F_W[m,k]|^2)^2}$$

- Maximizing this will result in sparsity
  - Which minimizes blurring on time and frequency axes

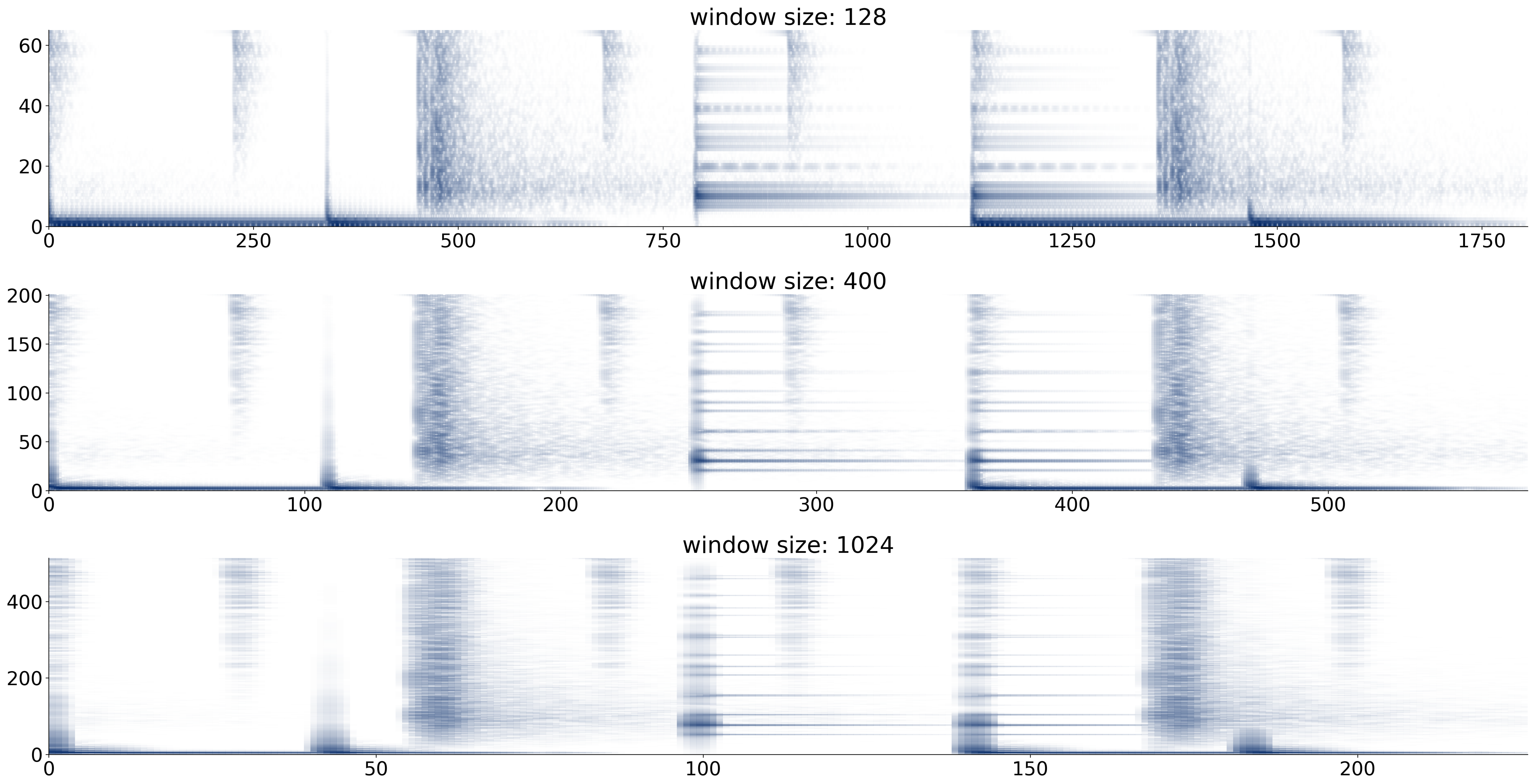
Optimizing window size

- How to optimize w.r.t discrete DFT size N?
  - Introduce a continuous proxy!
- Optimize loss w.r.t. an analysis window spread σ

$$W_m[n] = \exp\left[-\left(\frac{n}{2\sigma}\right)^2\right]$$

- Then infer transform size as $N = \lfloor 6\sigma \rfloor$
  - σ is continuous and can easily be optimized via gradient descent

Example on drum sound



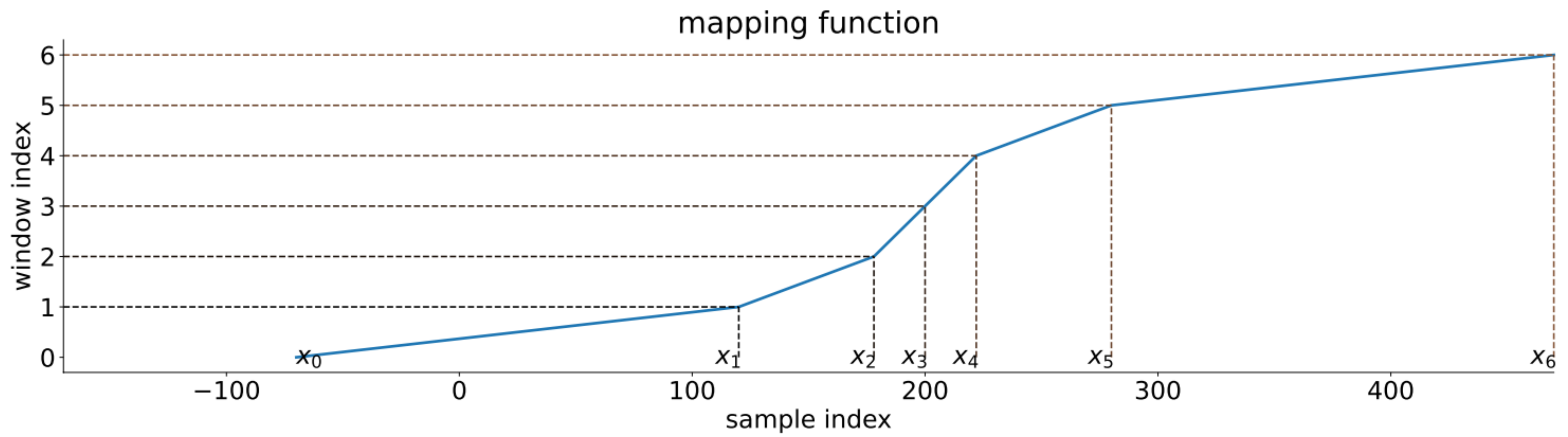
- Select optimal window length by optimizing sparsity (instead of eyeballing)

Optimizing size dynamically

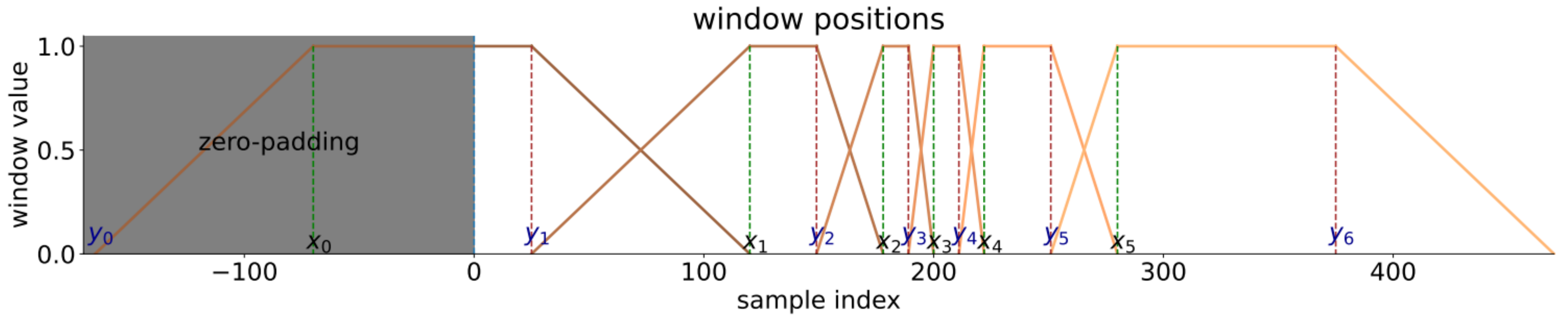
- We often need to change the window size over time
  - E.g. when the input sound varies from slow to fast
- More complex optimization problem
  - We need to adjust hop size and window accordingly
- We use trapezoidal window formulation
  - We estimate the start and end of trapezoidal windows

Dynamically varying size and hop?

- Trapezoidal window
  - Transform and hop size determined by the trapezoid shape
- Monotonic Mapping from window index to audio sample index



$$\mathcal{T}(m, x_i, y_i, x_{i+1}, y_{i+1}) = \begin{cases} \frac{m-y_i}{x_i-y_i} & \text{if } y_i \leq m < x_i \\ 1 & \text{if } x_i \leq m < y_{i+1} \\ 1 - \frac{m-y_{i+1}}{x_{i+1}-y_{i+1}} & \text{if } y_{i+1} \leq m < x_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

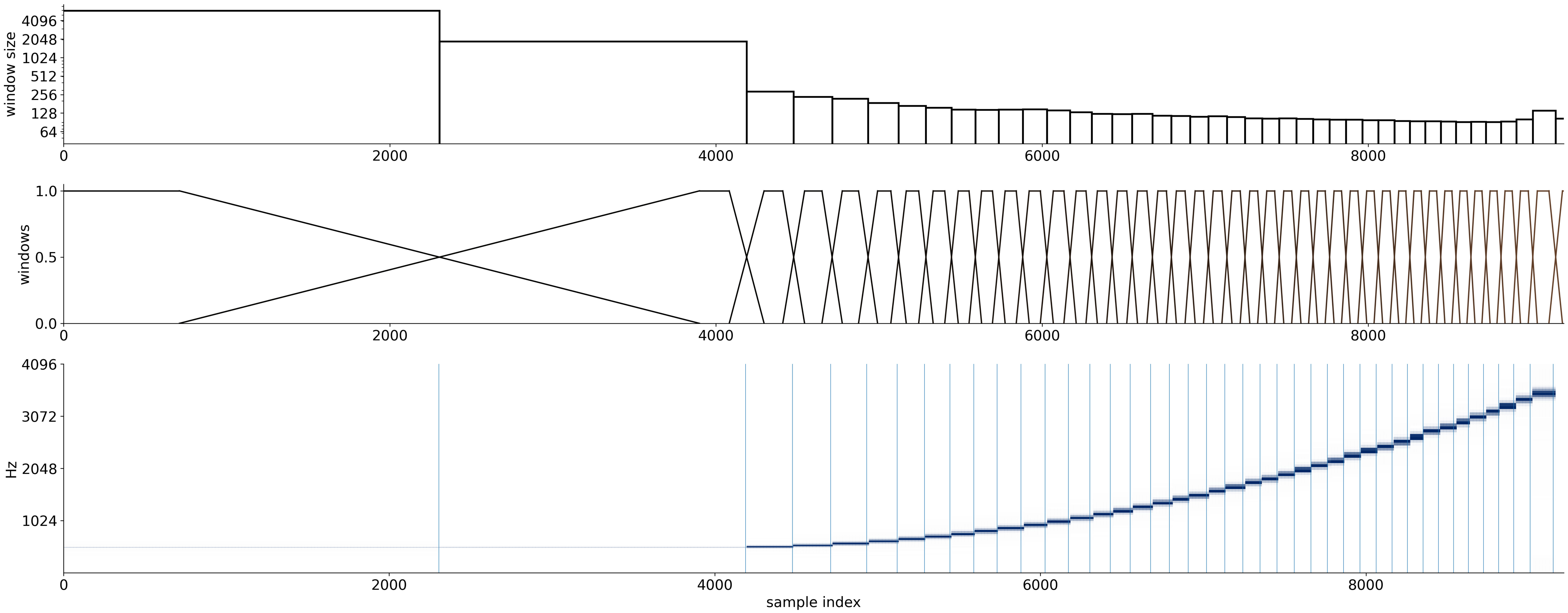


Parametric model for window placement

- Unconstrained Monotonic Neural Nets (UMNNs) can be used for differentiable mapping
  - Determines window distribution across audio
- Loss computed as sum of Kurtosis across each windowed audio segment $\mathcal{L}_q = -\sum_{\forall m} C[m, W]$
- UMNN weights updated via GD to obtain optimal window distribution that maximizes sparsity

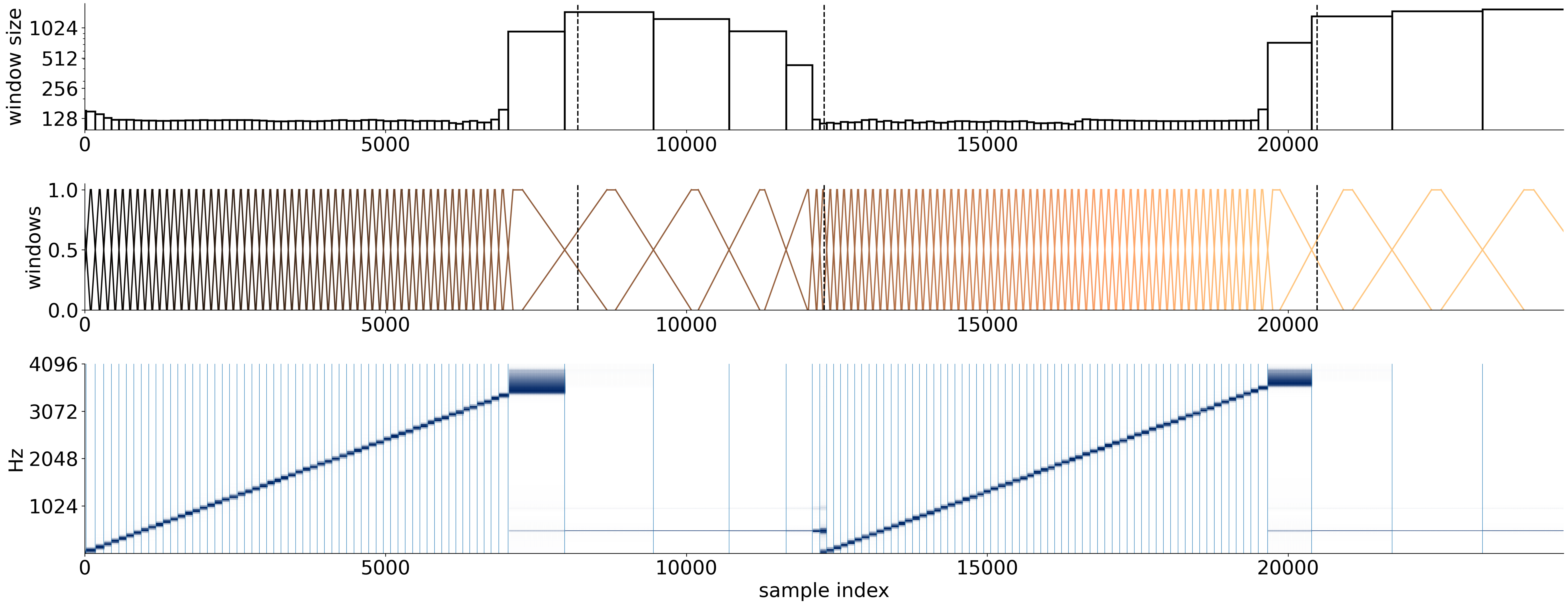
Example on exponential chirp

- Optimizing for sparsity again
  - As chirp progresses window size shrinks to reduce freq. blurring



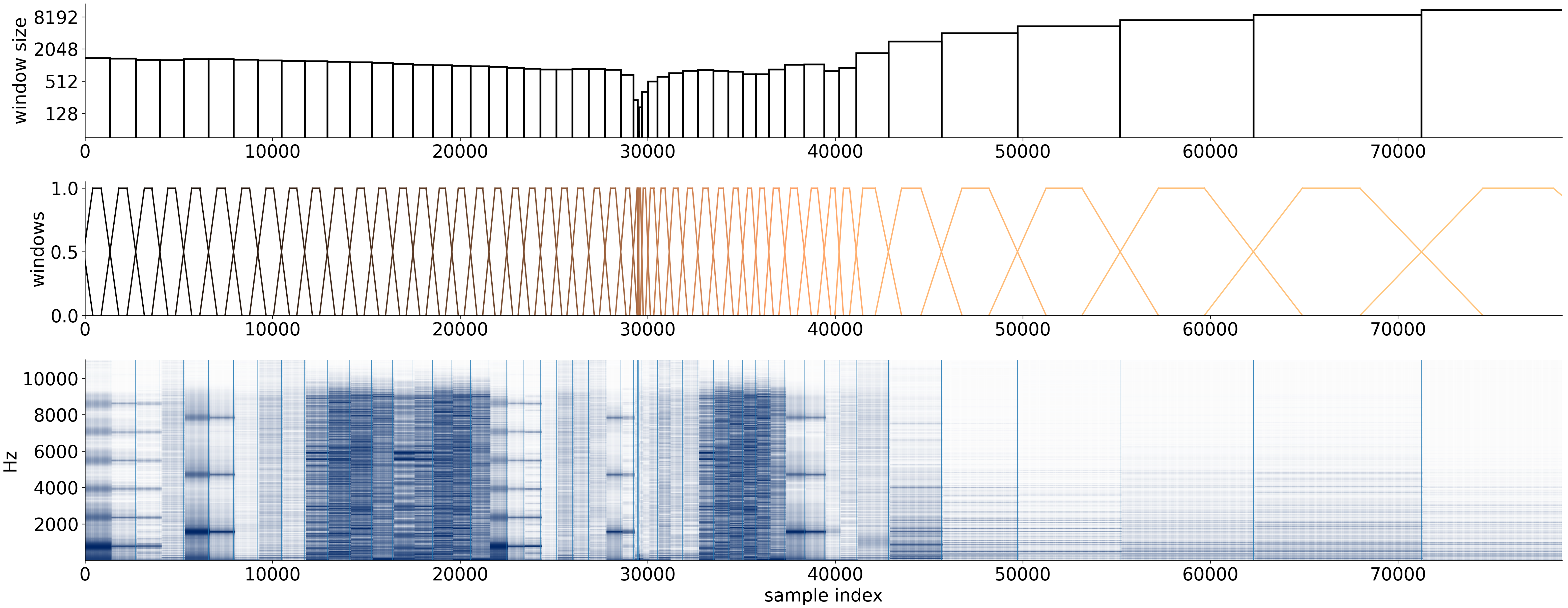
Linear chirps and steady tones

- Constant window for chirp, large windows for tones



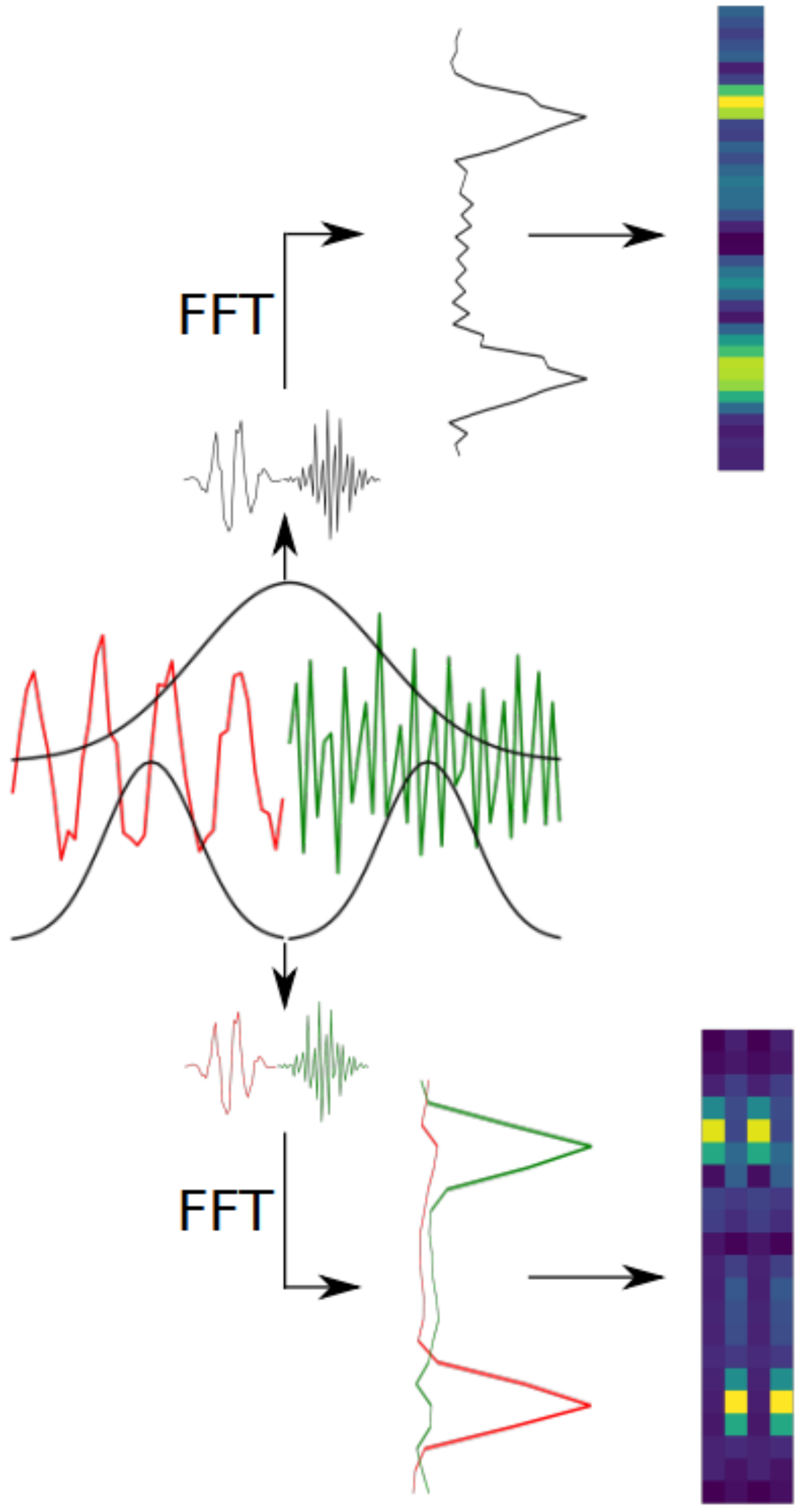
Drums and slow piano notes example

- Short windows during drums, longer during piano



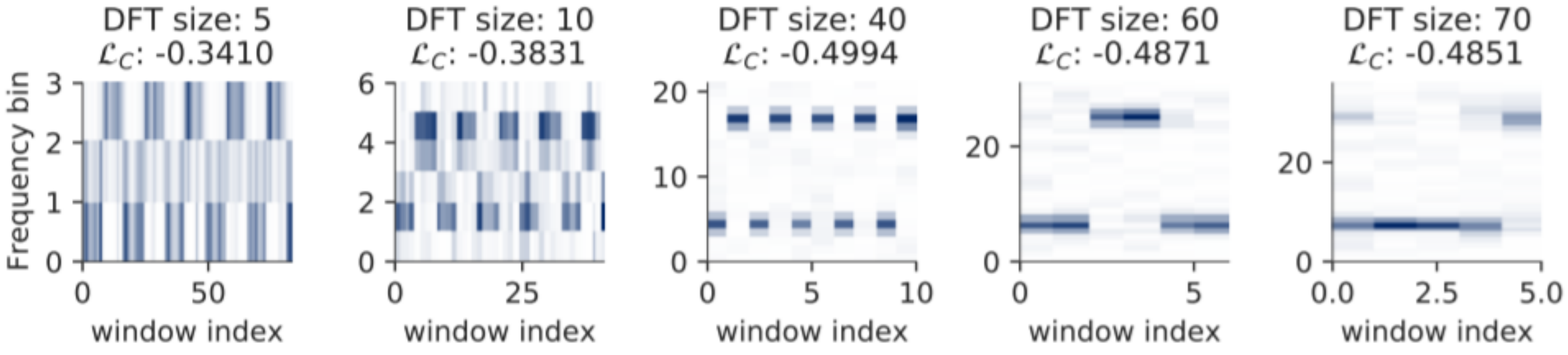
What about other cost functions

- What if we want to optimize w.r.t a specific downstream task like classification?



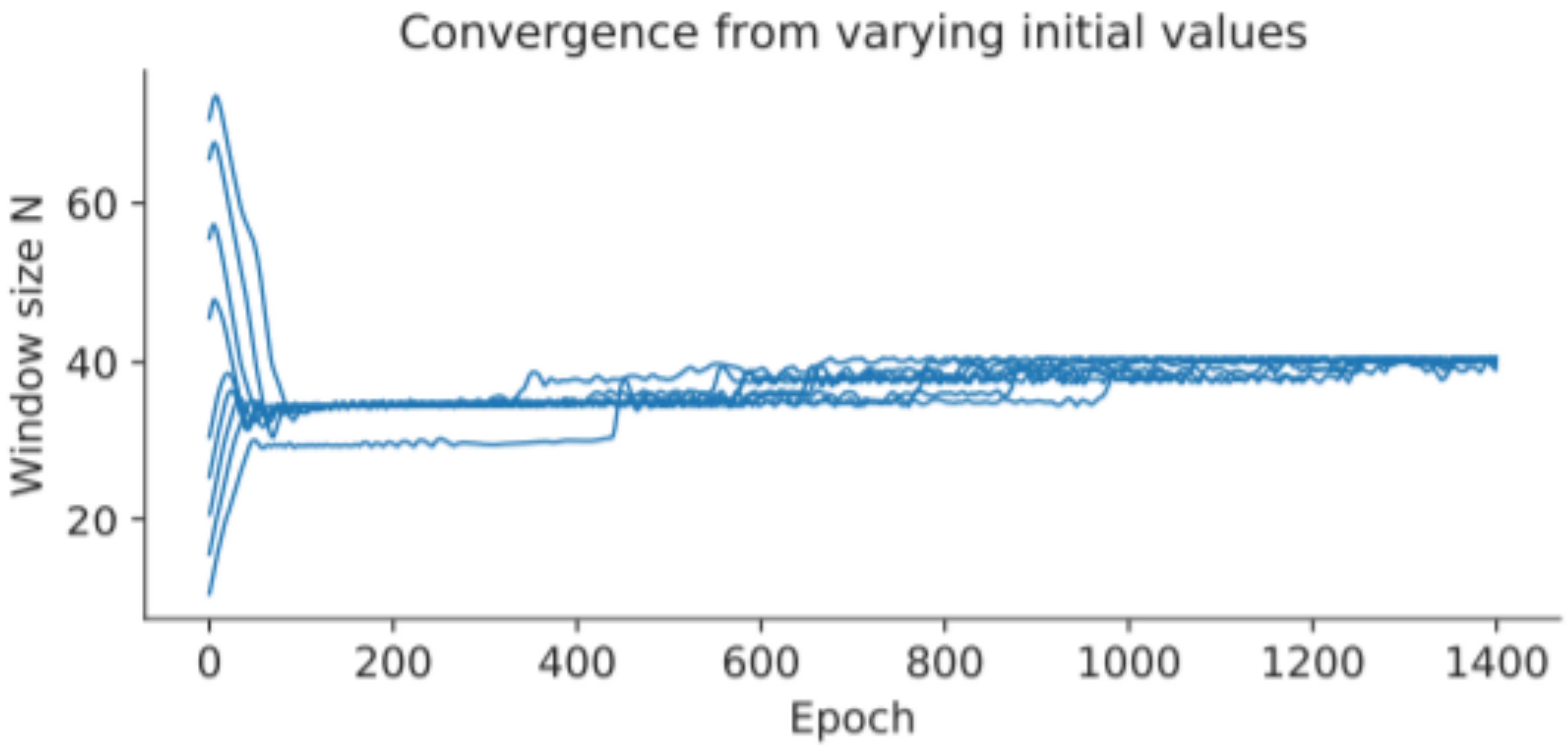
Classification Loss *Penalize Small Windows*

$$\mathcal{L}_C = \boxed{-\sum_{\forall m}\sum_{i} t_i[m]\log(z_i[m])} + \boxed{\frac{\lambda}{\sigma}}$$



Differentiable Search over Grid Search!

- Obtain final "optimal" window length differentiably
  - Significant speedup by avoiding search over many points
- Can be integrated inside larger processing system
  - E.g. a classification neural net



Conclusions

- Optimizing STFT parameters using gradient descent
  - Allows auto-tuning of STFT inside larger neural net pipelines
  - More efficient alternative to expensive grid search
- Optimization can be done w.r.t. arbitrary tasks
  - Can be used for a variety of methods
- Methodology useful for other integer parameters