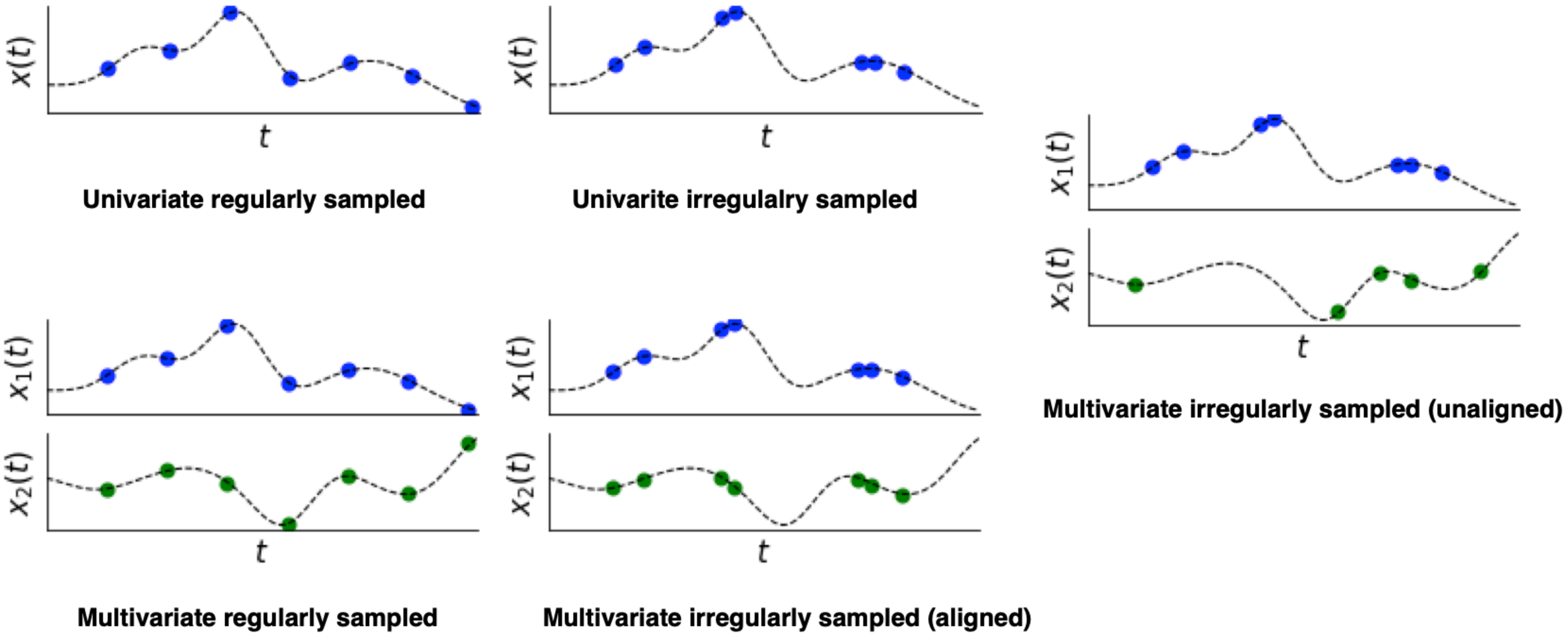
ECE209AS (Winter 2021)

Lecture 5: Learning with Irregularly Sampled Time Series Data

Mani Srivastava
mbs@ucla.edu
Networked & Embedded Systems Lab
ECE & CS Departments
UCLA

Regularly and Irregularly Sampled Time Series



Can we learn classification models on irregularly sampled time series without prior imputation?
(imputation can be unreliable and sacrifice (which sacrifices interpretability)

https://arxiv.org/pdf/2012.00168.pdf

Why do we get Irregular Sampling?

Why do we get Irregular Sampling?

- Availability of sensors
  - energy, mobility, multi-tenancy, human operator

Why do we get Irregular Sampling?

- Availability of sensors
  - energy, mobility, multi-tenancy, human operator
- Samples are lost or dropped
  - network outage, bit corruption

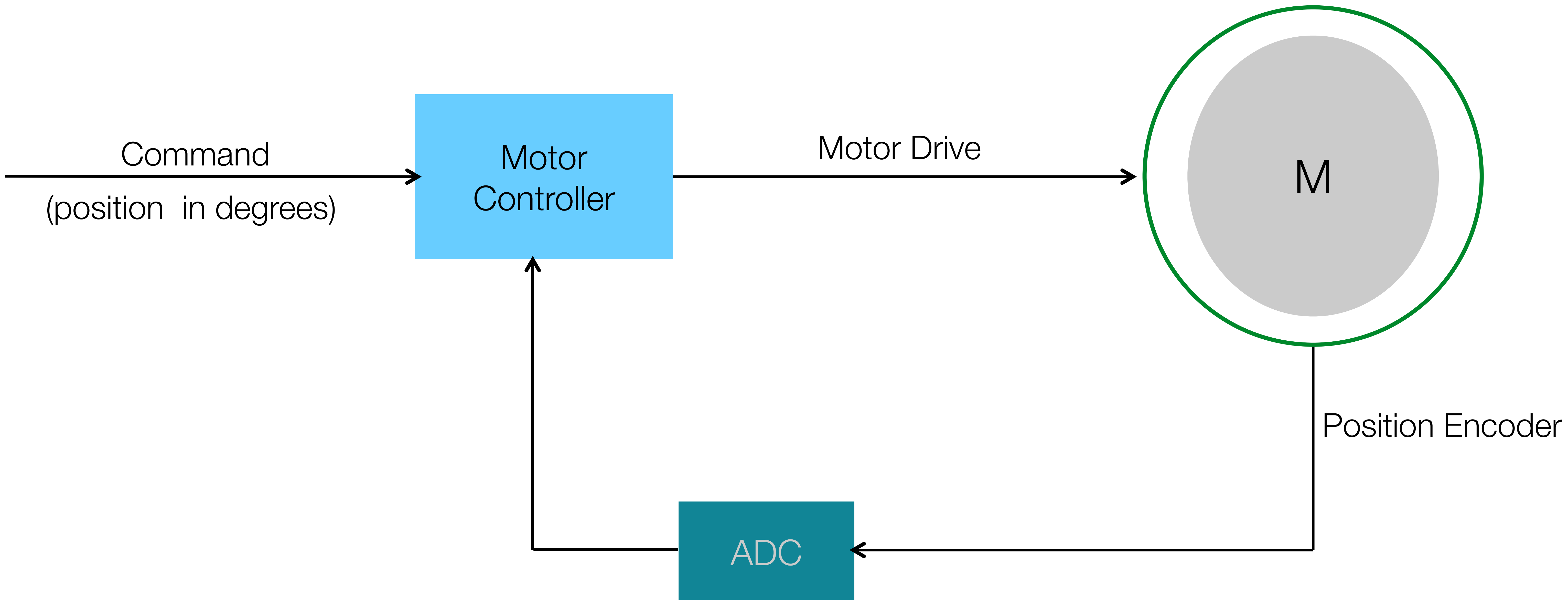
Why do we get Irregular Sampling?

- Availability of sensors
  - energy, mobility, multi-tenancy, human operator
- Samples are lost or dropped
  - network outage, bit corruption
- Sensors report asynchronous events
  - e.g. motion sensors, event-oriented imagers

Why do we get Irregular Sampling?

- Availability of sensors
  - energy, mobility, multi-tenancy, human operator
- Samples are lost or dropped
  - network outage, bit corruption
- Sensors report asynchronous events
  - e.g. motion sensors, event-oriented imagers
- Sensors sampling intervals are adapted
  - state of the process being sampled (e.g. patient's health)

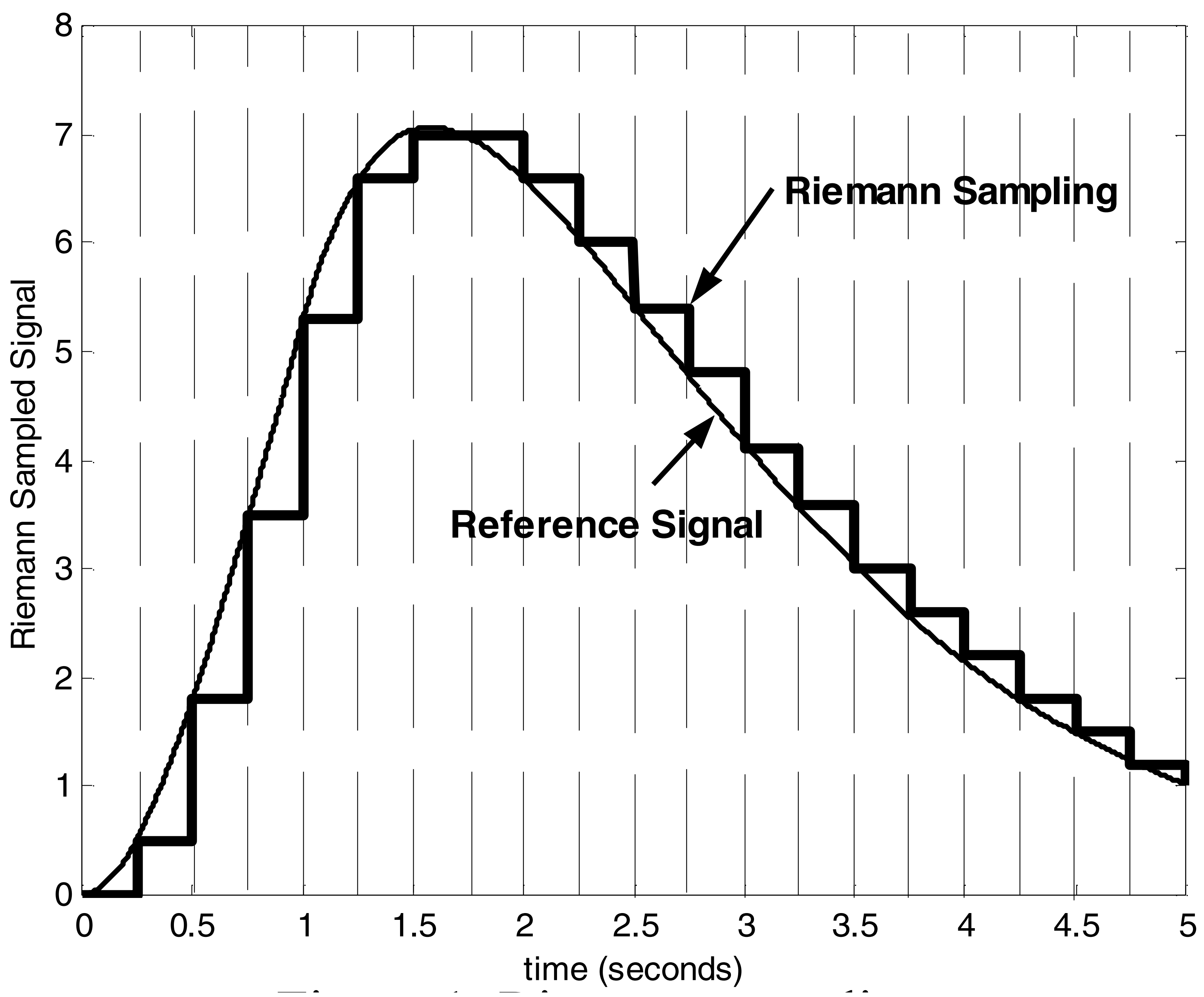
Sampling Adaptively

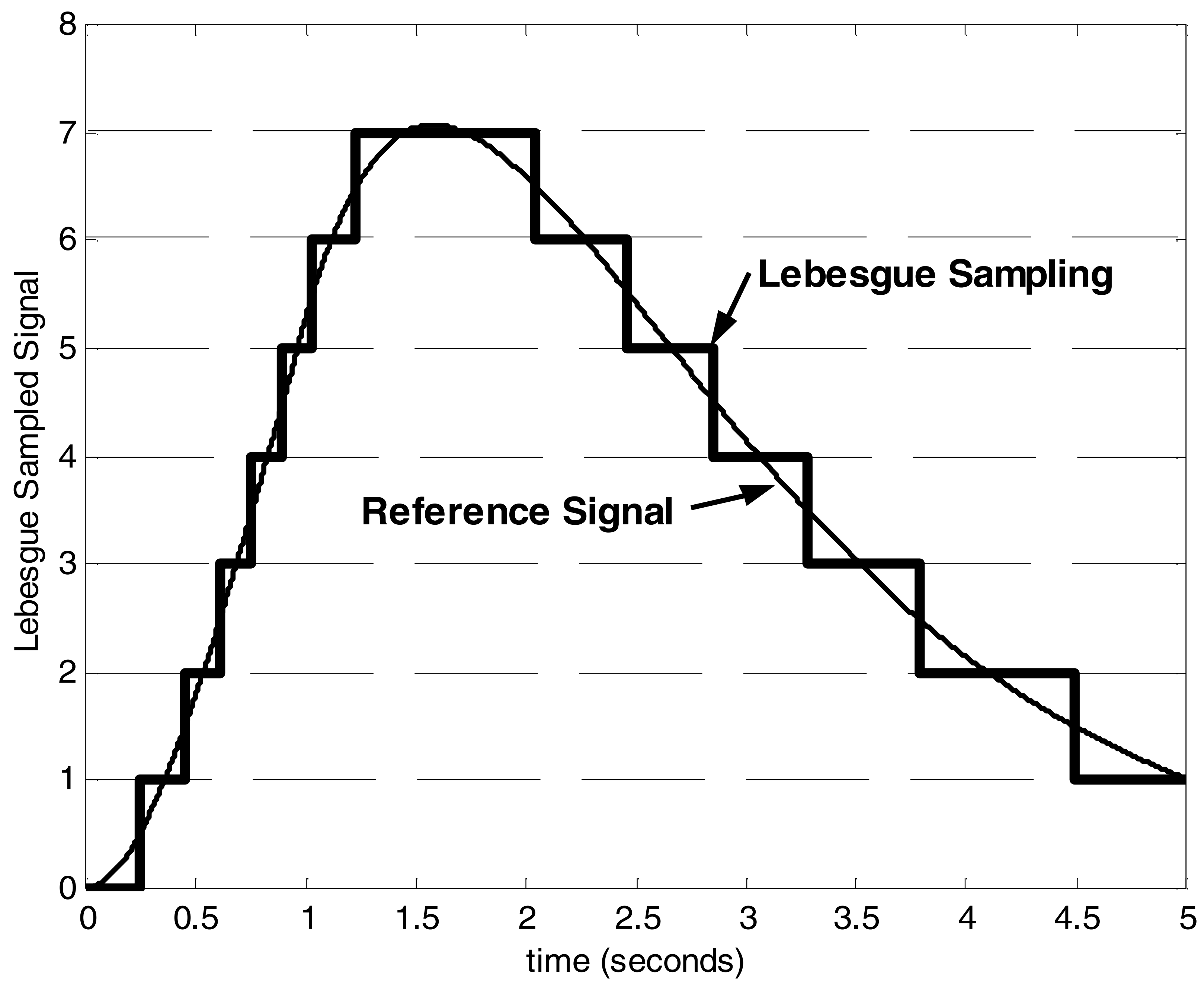


- Idea: Instead of sampling periodically as is traditionally done, could we sample only when the position of the motor changes sufficiently?
  - sample more frequently if position changes more rapidly

- This sampling policy is called *Lebesgue* Sampling

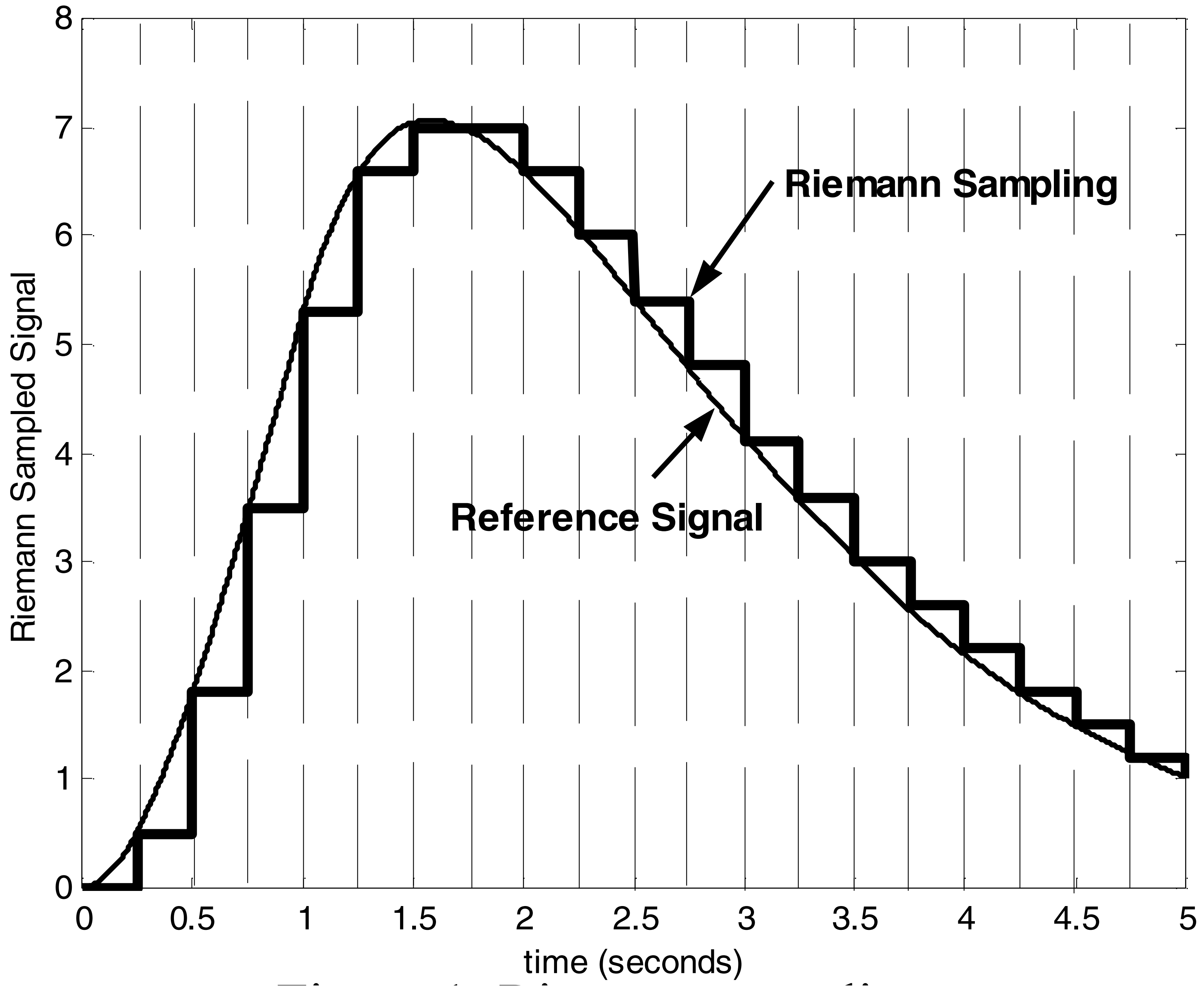
Riemann Sampling vs. Lebesgue Sampling

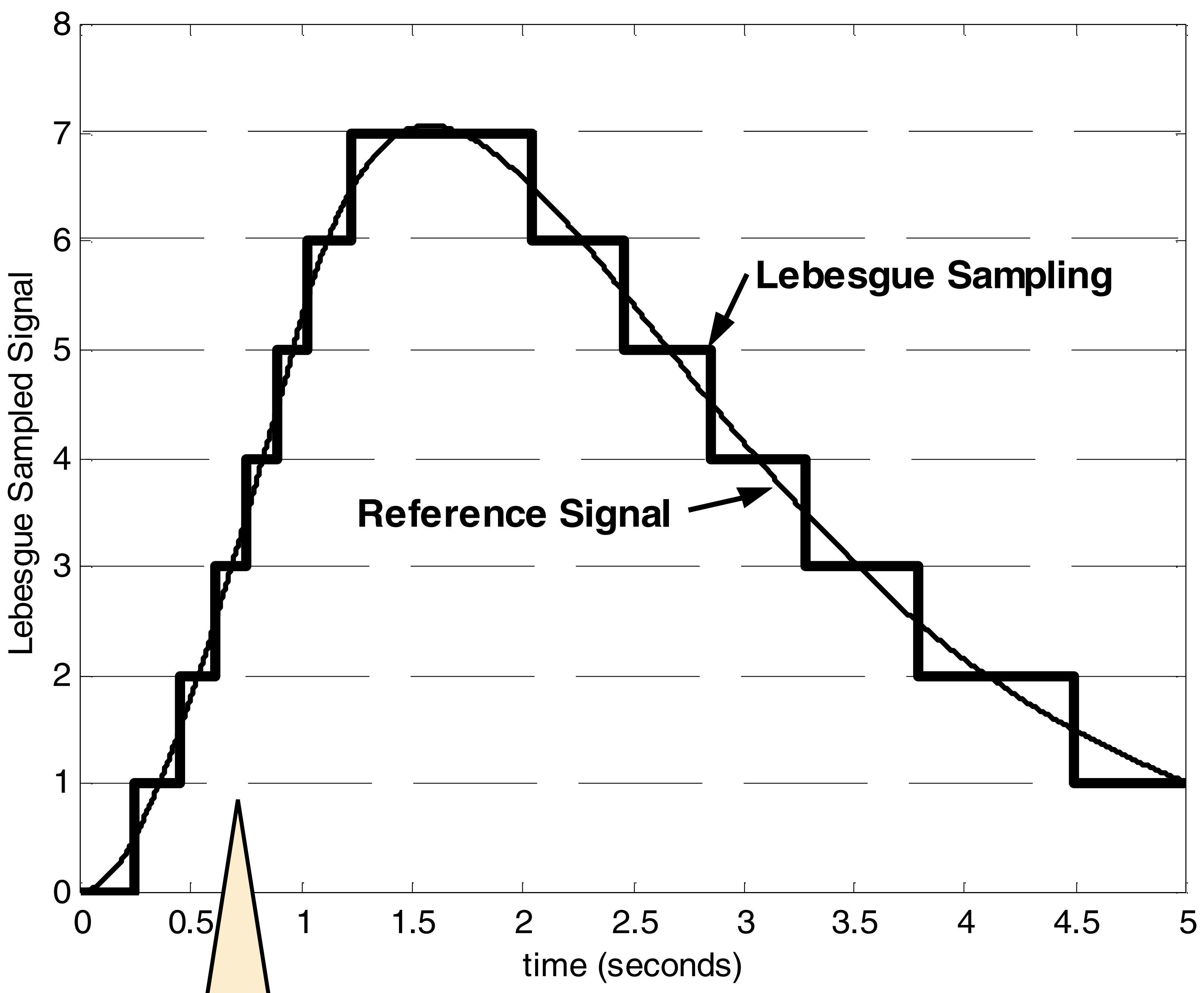




From:Roy McCann, Anil Kumar Gunda, Suchit Reddy Damugatla, "Improved Operation of Networked Control Systems using Lebesgue Sampling"

Riemann Sampling vs. Lebesgue Sampling

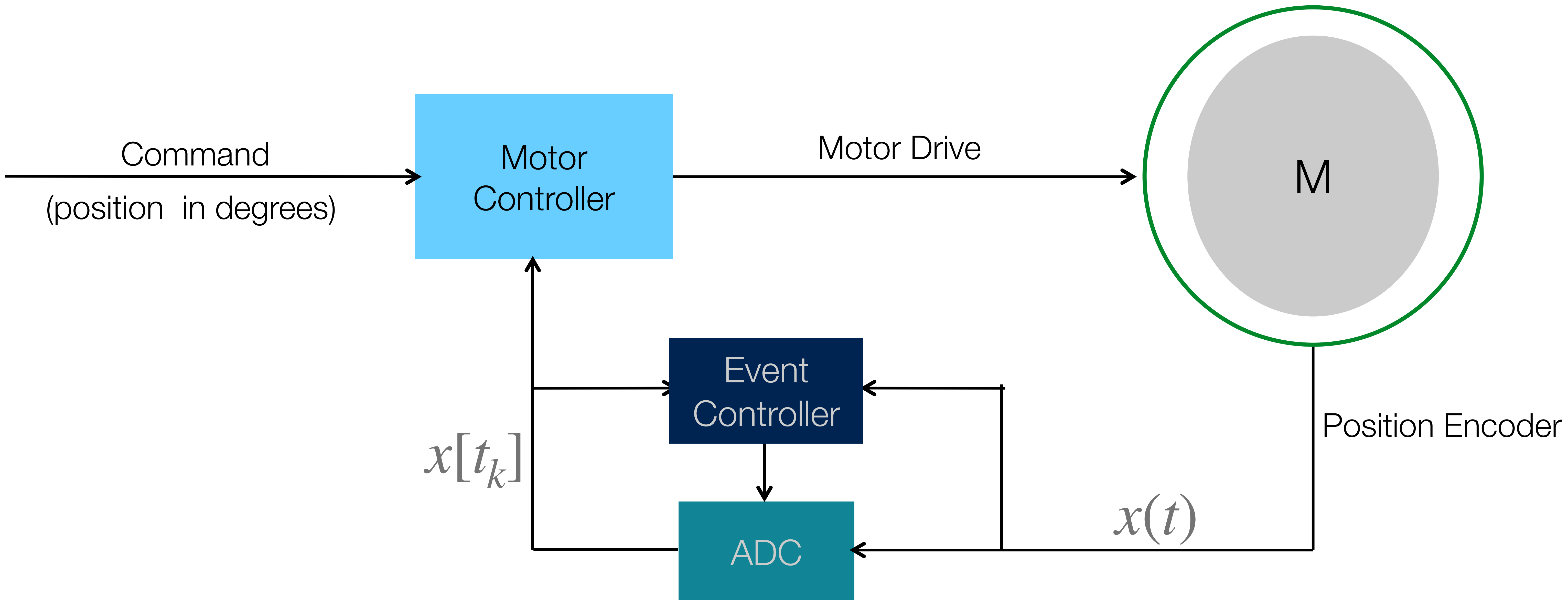




From:Roy McCann, An... Improved
Operation of Netw... ing”

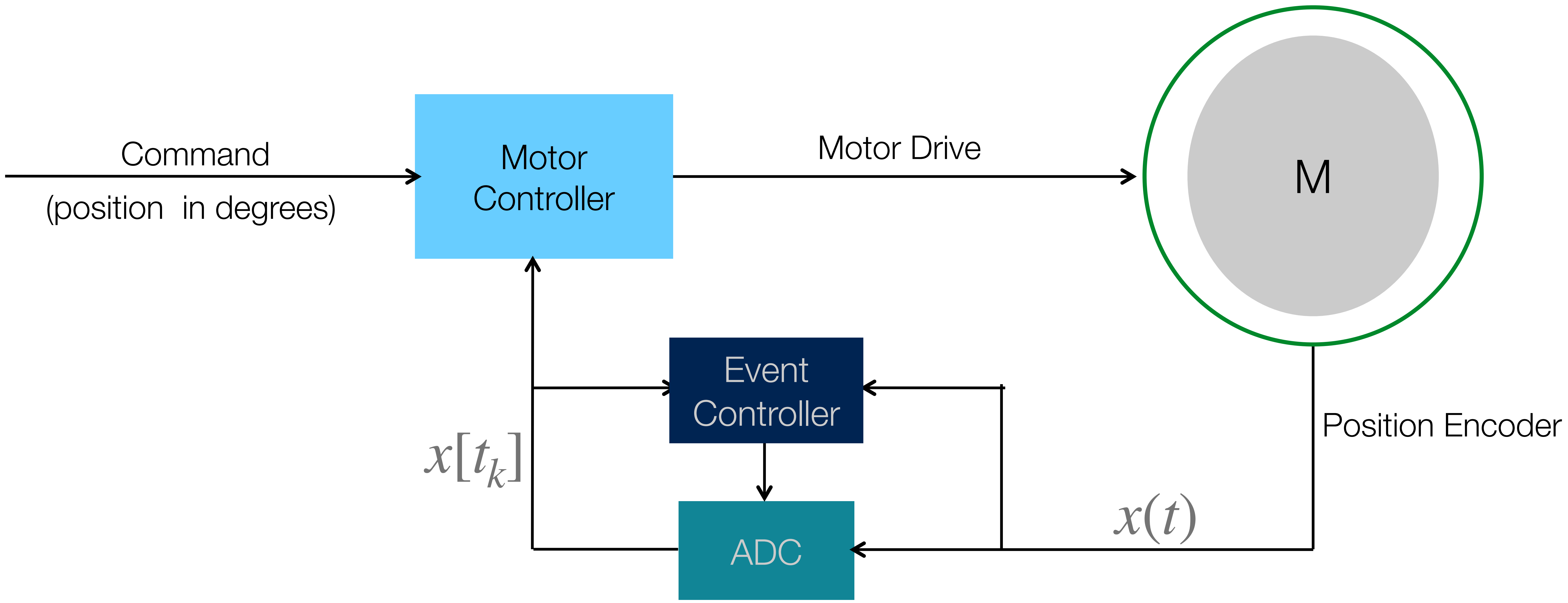
How do we decide the Lebesgue interval?

Event-Triggered Control System



- Introduce an Event Controller that monitors the Position Encoder signal continuously and determines the optimal sampling time
- $x(t)$ is the continuous-time analog position encoder signal
- $x[t_k]$ is the last sample produced by the ADC

Event-Triggered Control System



- **Theorem [Anta-Tabuada]:** Sampling is optimal and the system is guaranteed stability if the Event Controller produces a sampling trigger whenever:

$$||x(t) - x[t_k]|| > \sigma||x(t)||$$

  - σ is a design parameter that trades off average sampling rate and performance
  - Intuition: Lebesgue interval must be the “error” relative to size of signal

Self-Triggered Control System

- Event Controller needs to check the inequality at ALL times. Can one do better?
- Motor controller is designed with a model of the motor in mind
- Given the current state of the motor and the command input, control theory provides an estimate of the next state
- One can extend this estimation to predict when the state of the motor will violate the sampling inequality
- Thus, one can predict when the sample should be taken and schedule the ADC

$$CurrentMeasurement + SystemDynamics \Rightarrow NextSampleTime$$

- No continuous time check needed
- However, this does not work well when the model has uncertainties or when perturbations to the system cannot be bounded

Self-Triggered Control System

$$t_{k+1} = \alpha(\lambda, x[t_k])$$

Self-Triggered Control System

$$t_{k+1} = \alpha(\lambda, x[t_k])$$

Next transmission time

Self-Triggered Control System

$$t_{k+1} = \alpha(\lambda, x[t_k])$$

Next transmission time

Desired performance

Self-Triggered Control System

$$t_{k+1} = \alpha(\lambda, x[t_k])$$

Next transmission time

Desired performance

Measured state

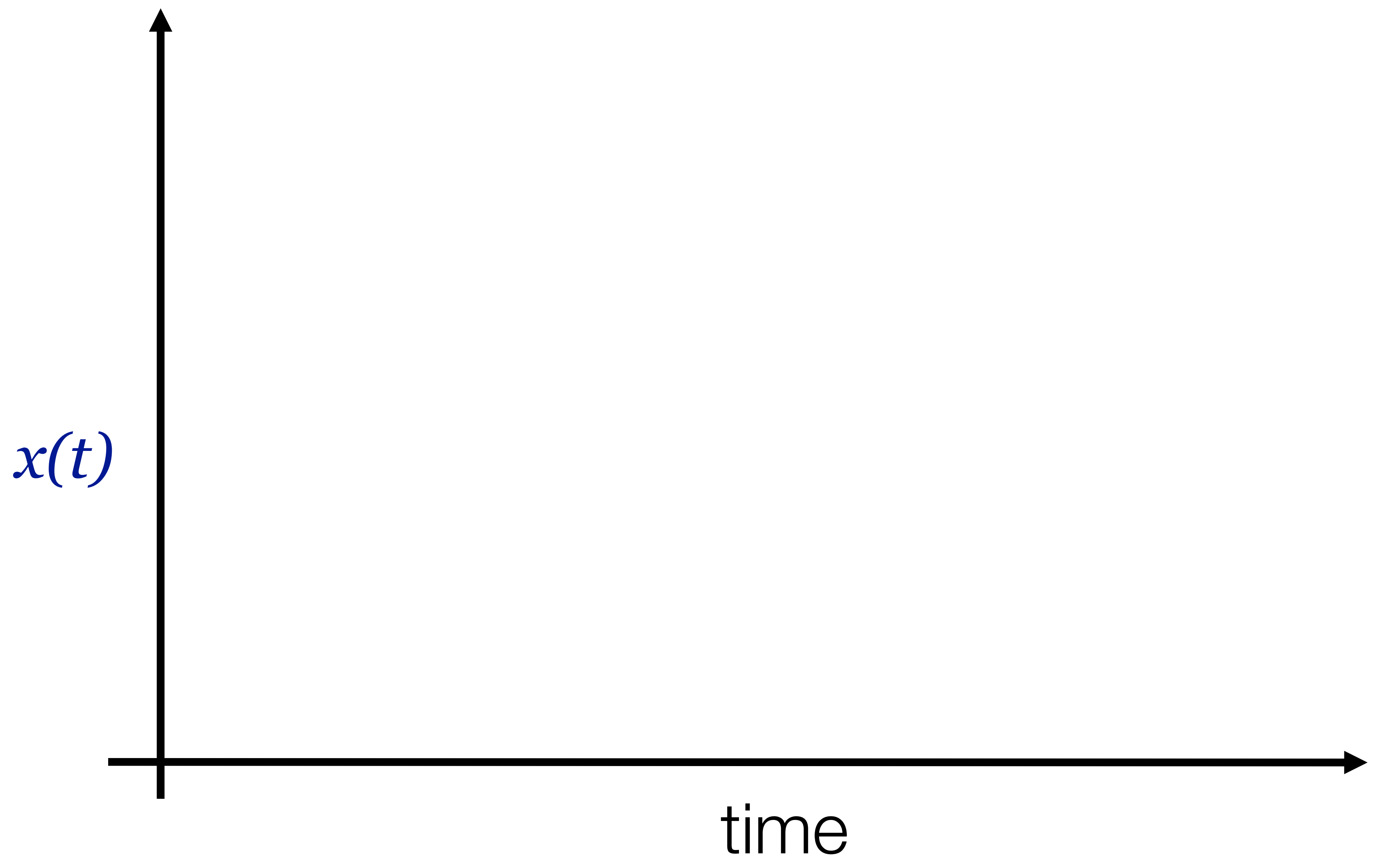
Self-Triggered Control System

$$t_{k+1} = \alpha(\lambda, x[t_k])$$

Next transmission time

Desired performance

Measured state



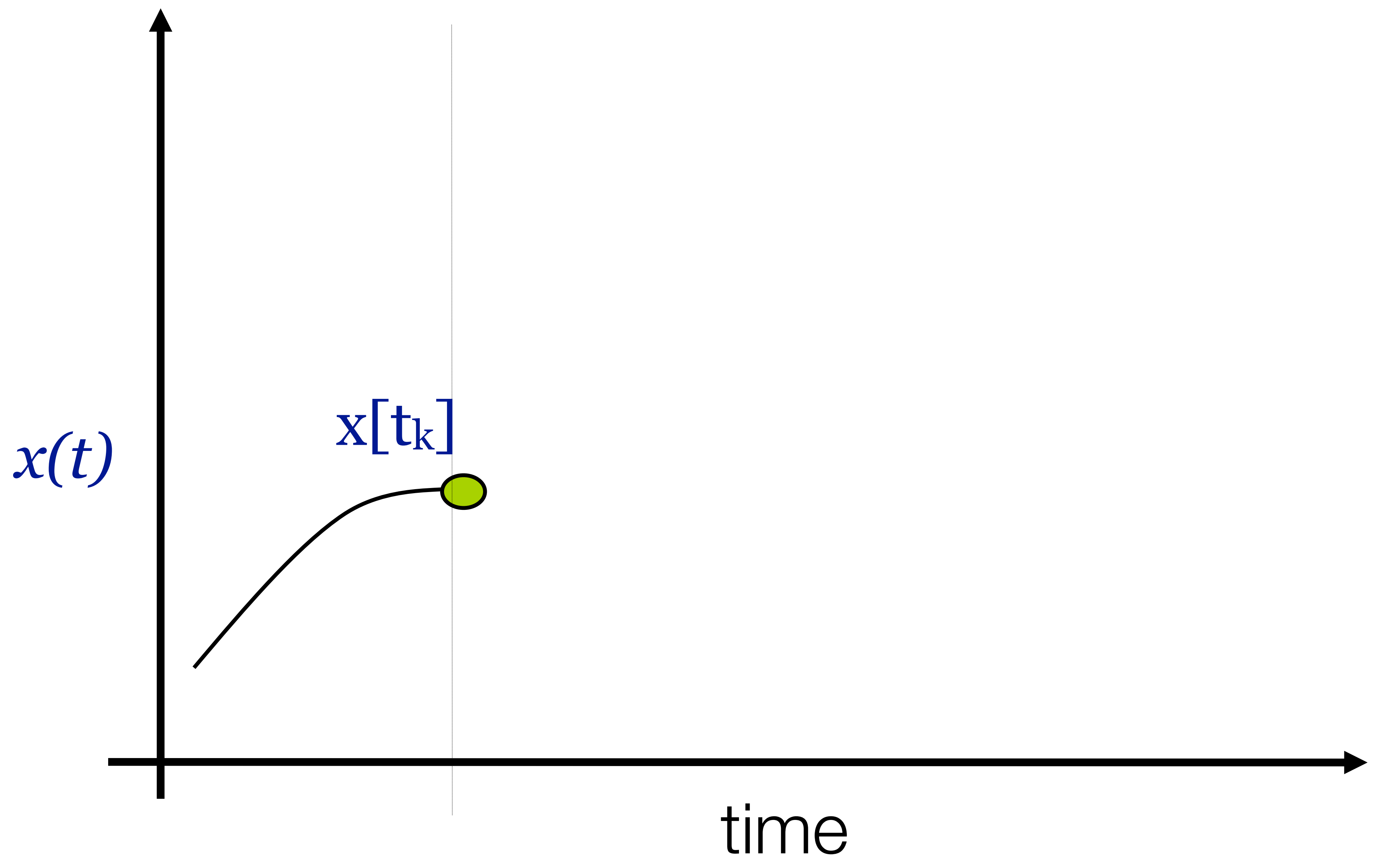
Self-Triggered Control System

$$t_{k+1} = \alpha(\lambda, x[t_k])$$

Next transmission time

Desired performance

Measured state



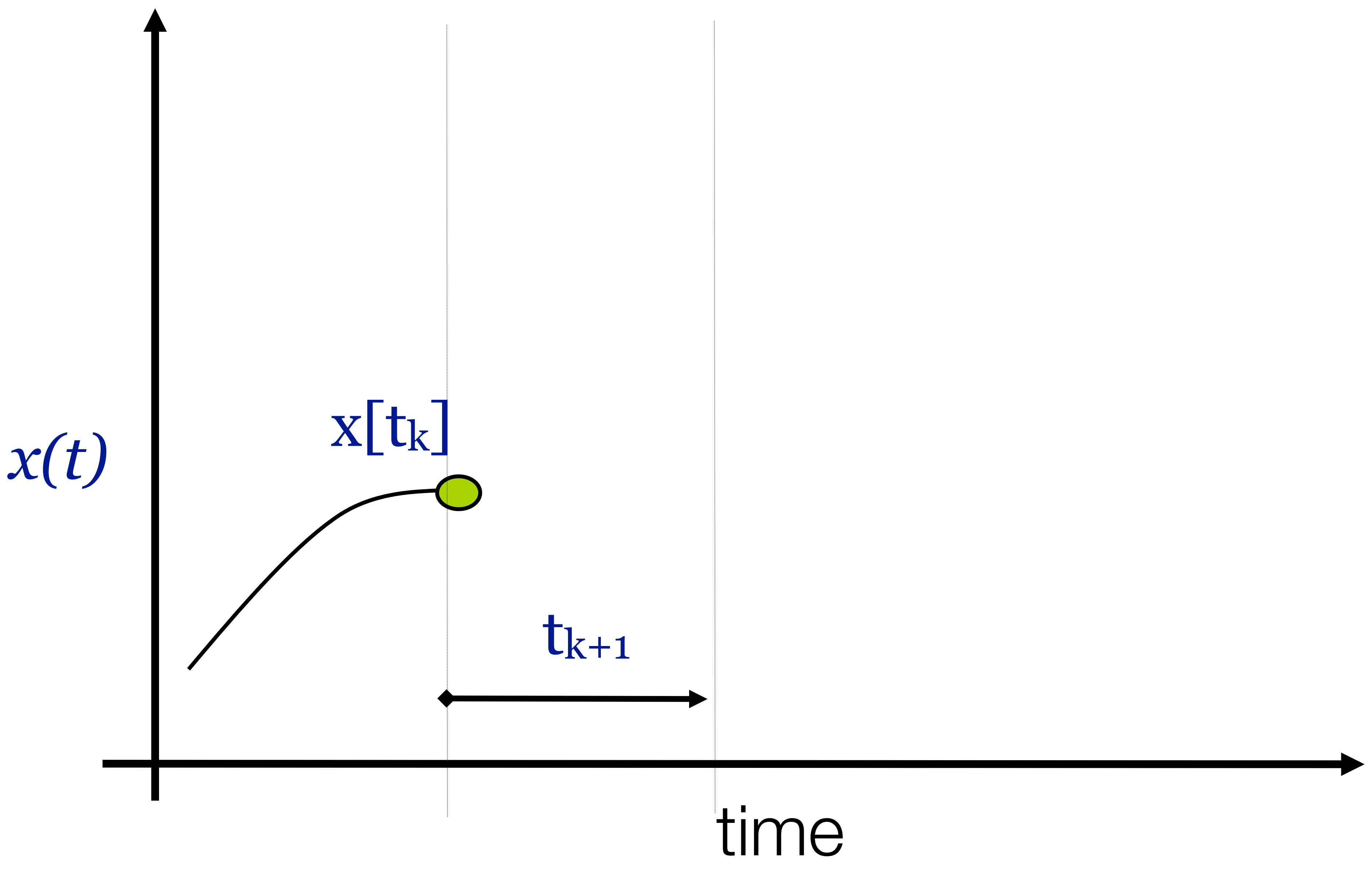
Self-Triggered Control System

$$t_{k+1} = \alpha(\lambda, x[t_k])$$

Next transmission time

Desired performance

Measured state



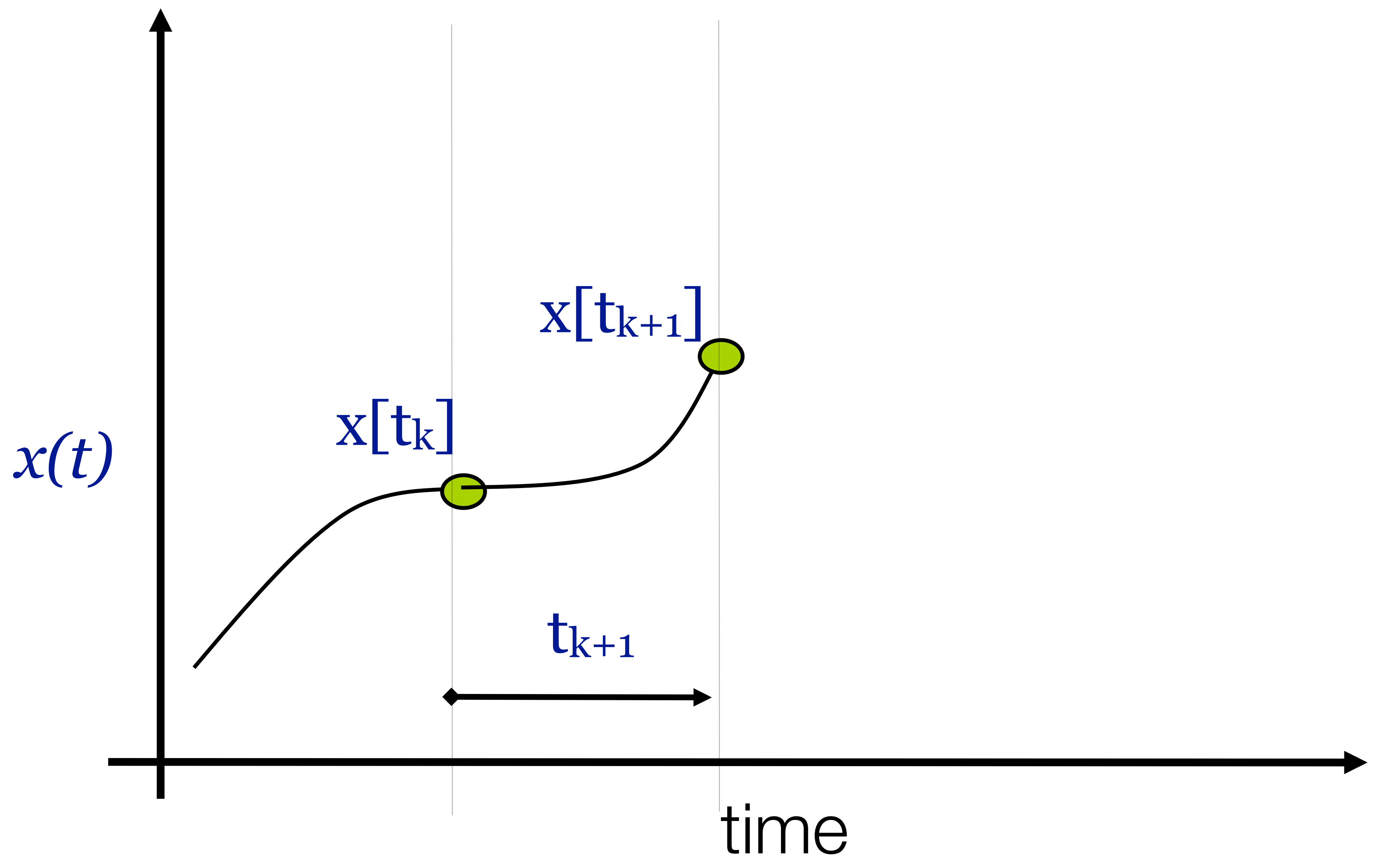
Self-Triggered Control System

$$t_{k+1} = \alpha(\lambda, x[t_k])$$

Next transmission time

Desired performance

Measured state



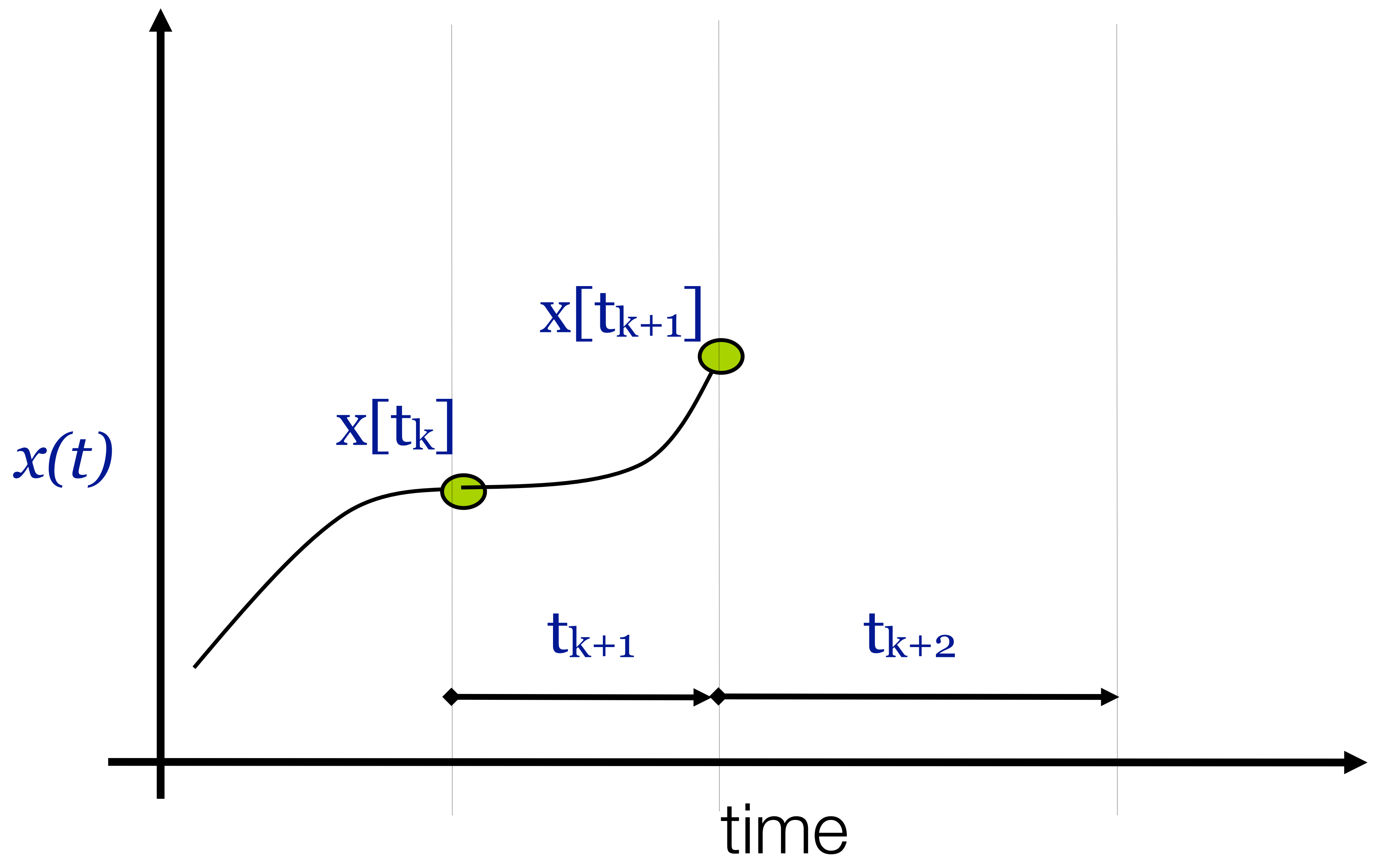
Self-Triggered Control System

$$t_{k+1} = \alpha(\lambda, x[t_k])$$

Next transmission time

Desired performance

Measured state



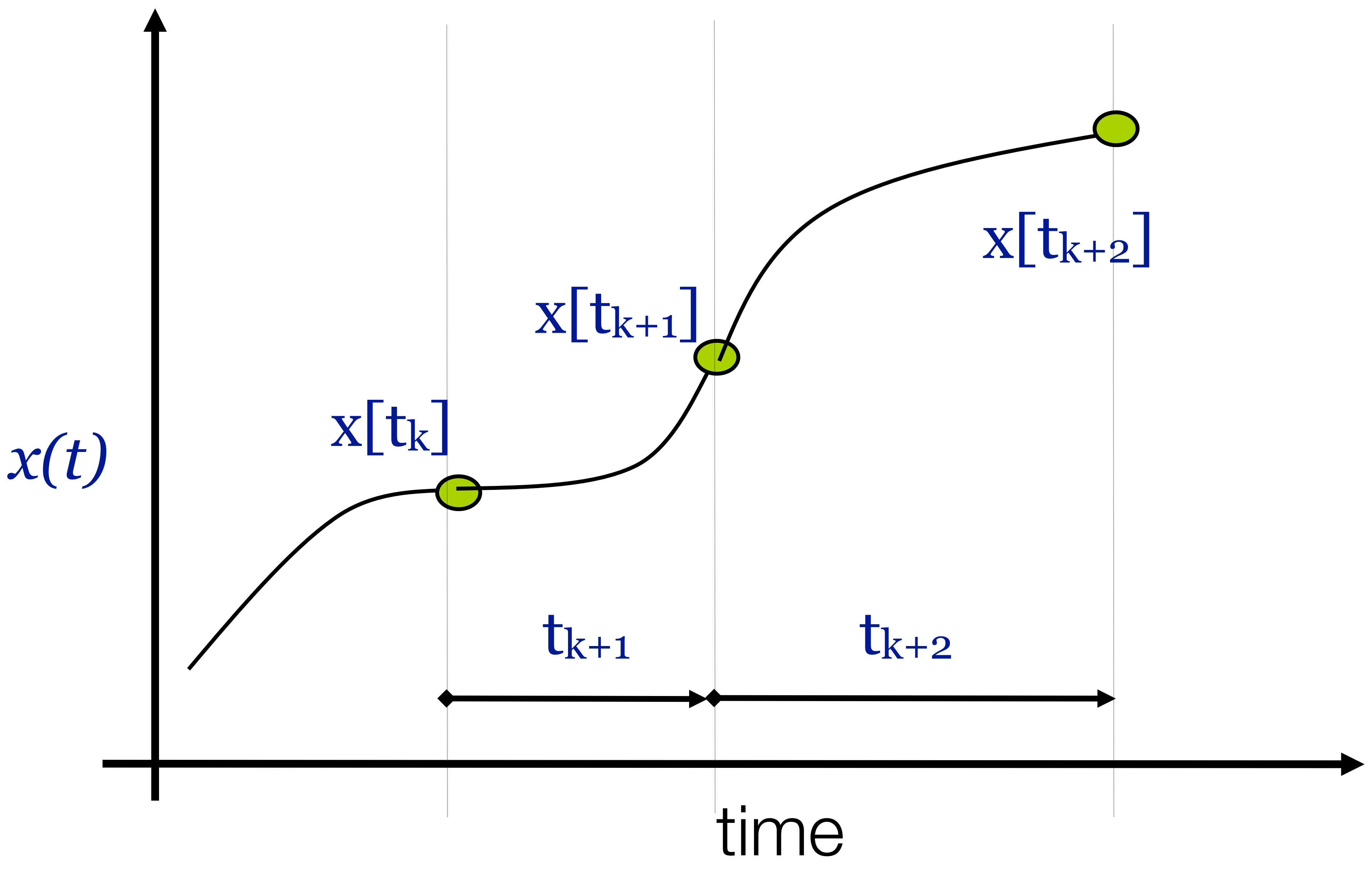
Self-Triggered Control System

$$t_{k+1} = \alpha(\lambda, x[t_k])$$

Next transmission time

Desired performance

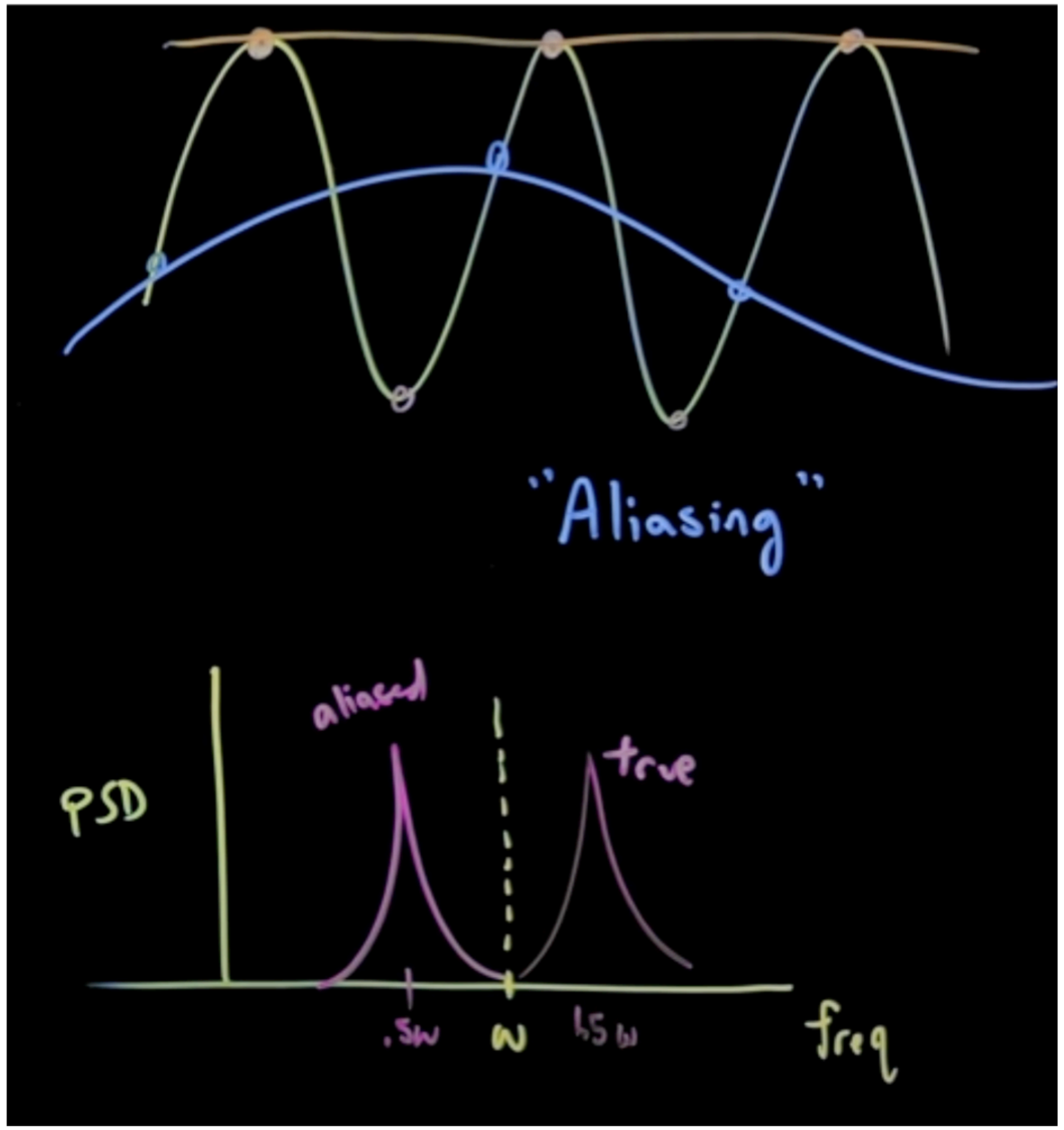
Measured state



Why do we get Irregular Sampling?

- Availability of sensors
  - energy, mobility, multi-tenancy, human operator
- Samples are lost or dropped
  - network outage, bit corruption
- Sensors report asynchronous events
  - e.g. motion sensors, event-oriented imagers
- Sensors sampling intervals are adapted
  - state of the process being sampled (e.g. patient's health)
- Compressive sampling of sensors

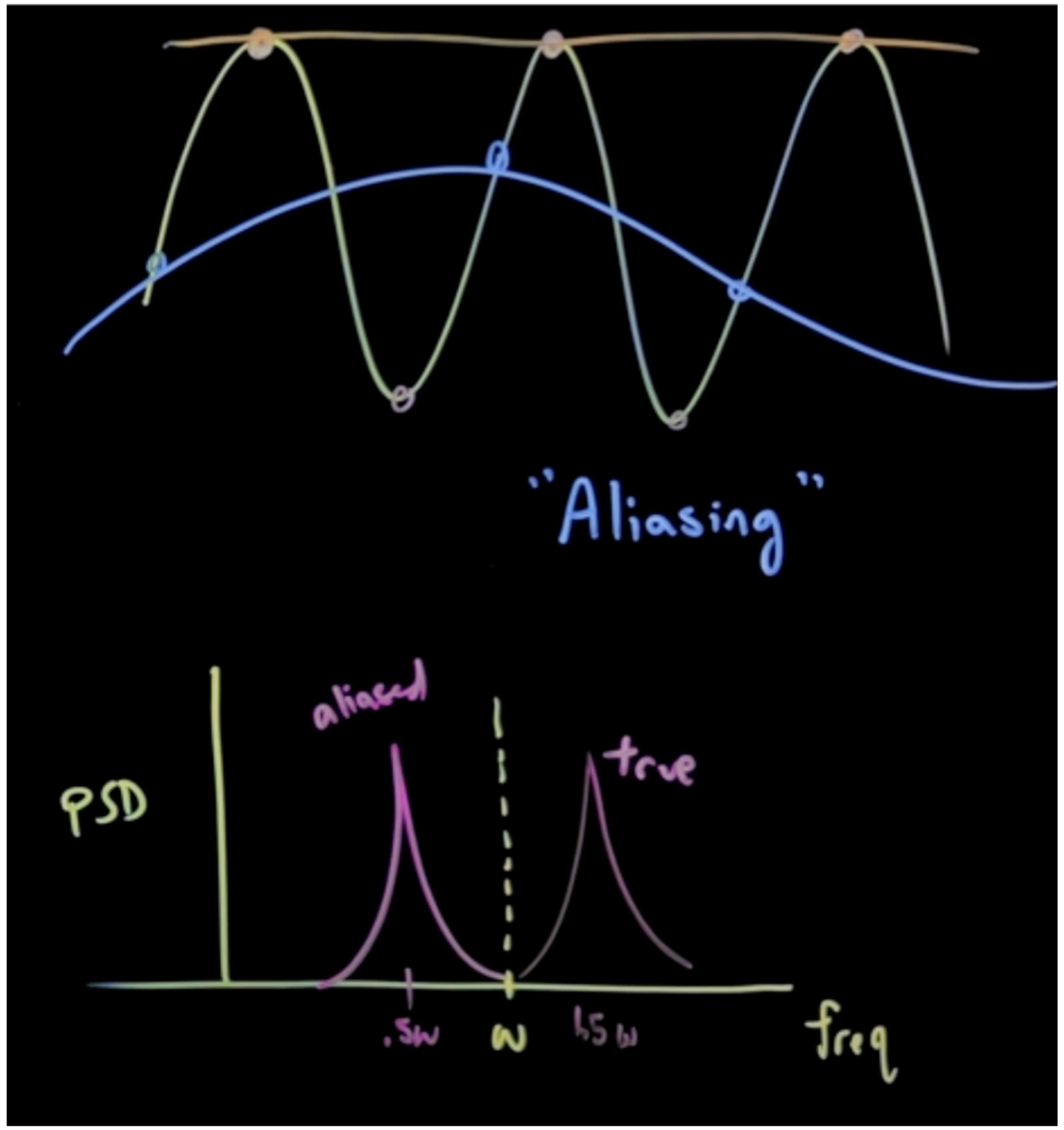
Sampling a Signal



Steve Brunton, "Shannon Nyquist Sampling Theorem"
https://www.youtube.com/watch?v=FcXZ28BX-xE&t=449s

Sampling a Signal

- Shannon-Nyquist Sampling Theorem:
  - To resolve all frequencies in a function, it must be sampled at twice the highest frequency present
  - A function containing no frequency > ω Hz is completely determined by sampling at 2ω Hz (Nyquist Rate) $\Delta t = \frac{1}{2\omega}$
  - Aliasing if we sample at a rate lower than 2ω

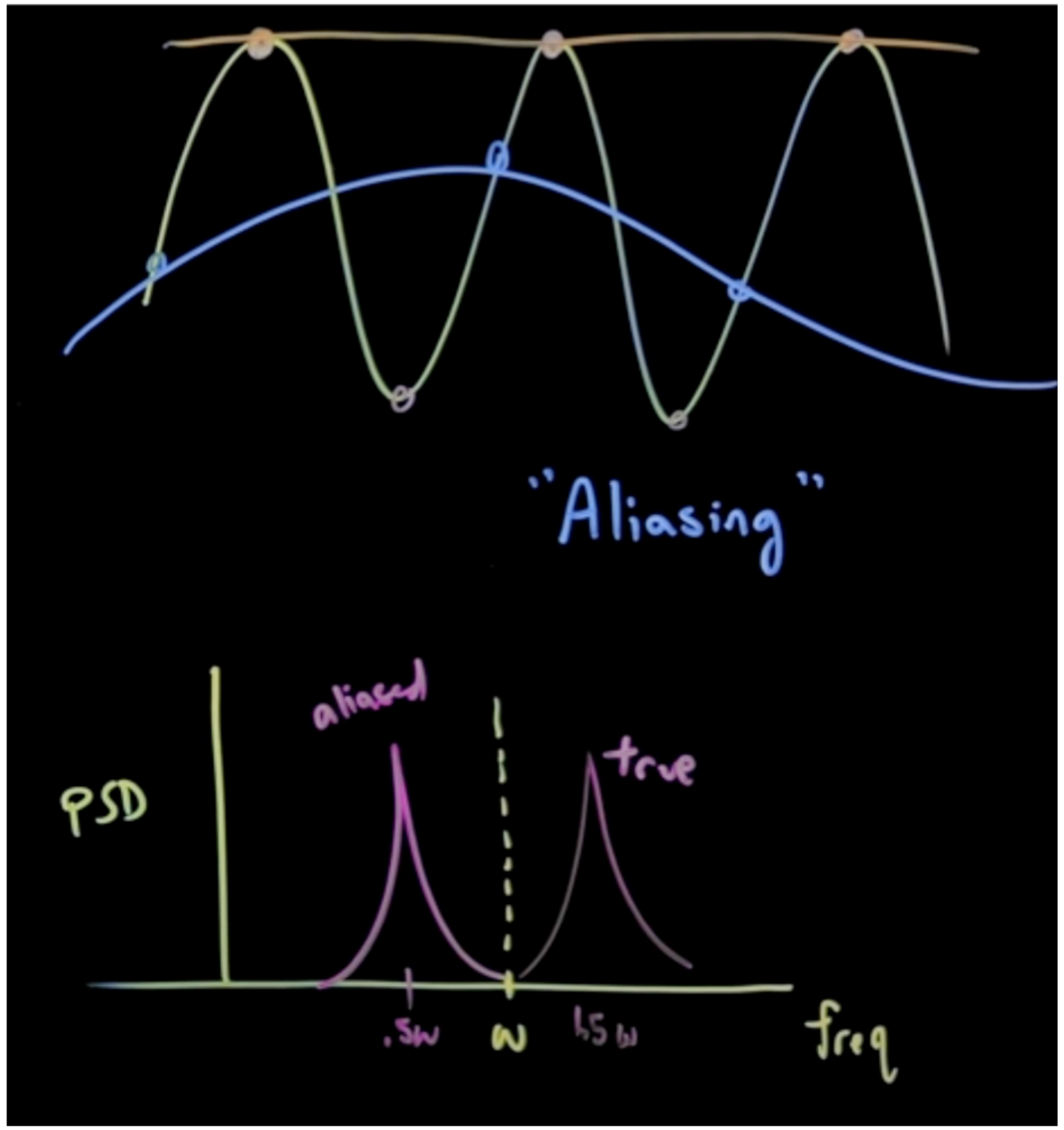


Steve Brunton, "Shannon Nyquist Sampling Theorem"
https://www.youtube.com/watch?v=FcXZ28BX-xE&t=449s

Sampling a Signal

- Shannon-Nyquist Sampling Theorem:
  - To resolve all frequencies in a function, it must be sampled at twice the highest frequency present
  - A function containing no frequency > ω Hz is completely determined by sampling at 2ω Hz (Nyquist Rate) $\Delta t = \frac{1}{2\omega}$
  - Aliasing if we sample at a rate lower than 2ω

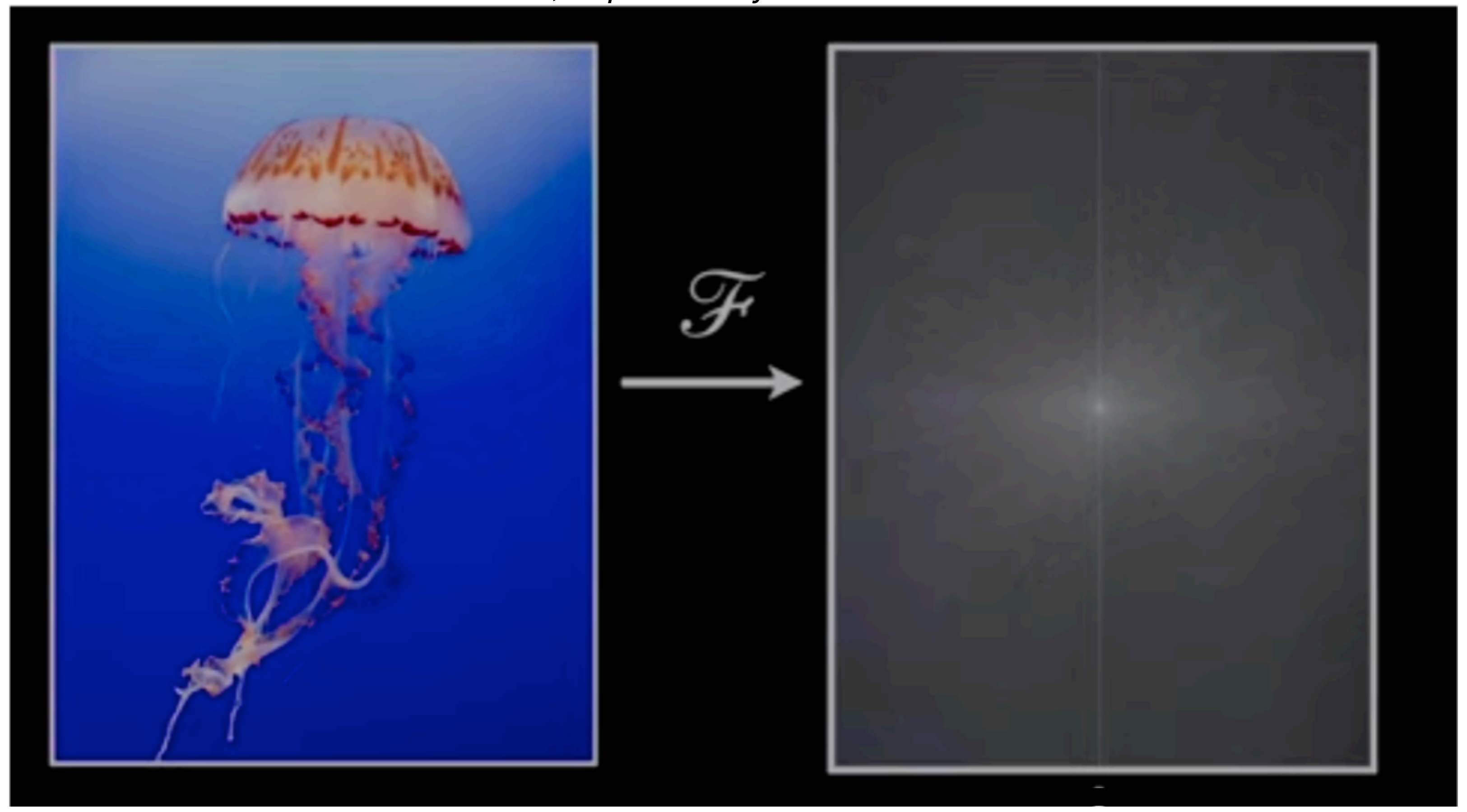
- Beating Shannon-Nyquist Sampling Theorem
  - Advances in applied mathematics, statistics, and optimization have changed how we thinks about sampling
  - Technically, the Shannon-Nyquist Sampling Theorem is necessary only signals that are broadband, i.e. densely packed with energy in all the frequencies from 0 to ω
  - But if the signal is sparse in frequency domain, one can beat the 2ω sampling rate requirement

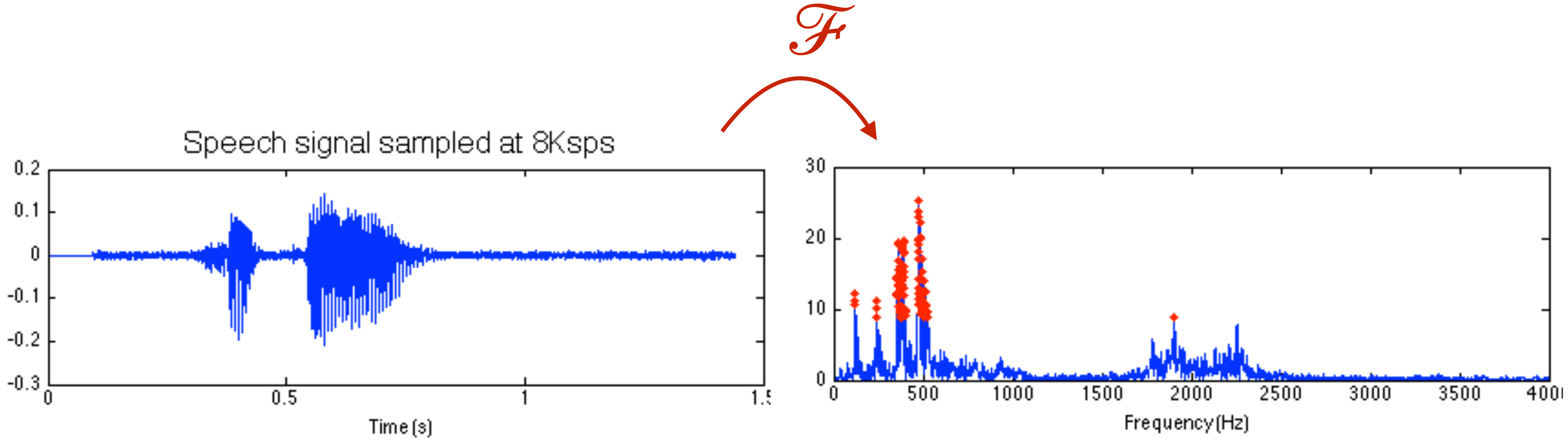


Steve Brunton, "Shannon Nyquist Sampling Theorem"
https://www.youtube.com/watch?v=FcXZ28BX-xE&t=449s

Motivating Compressive Sampling

Steve Brunton, https://www.youtube.com/watch?v=hmBTACBGWJs

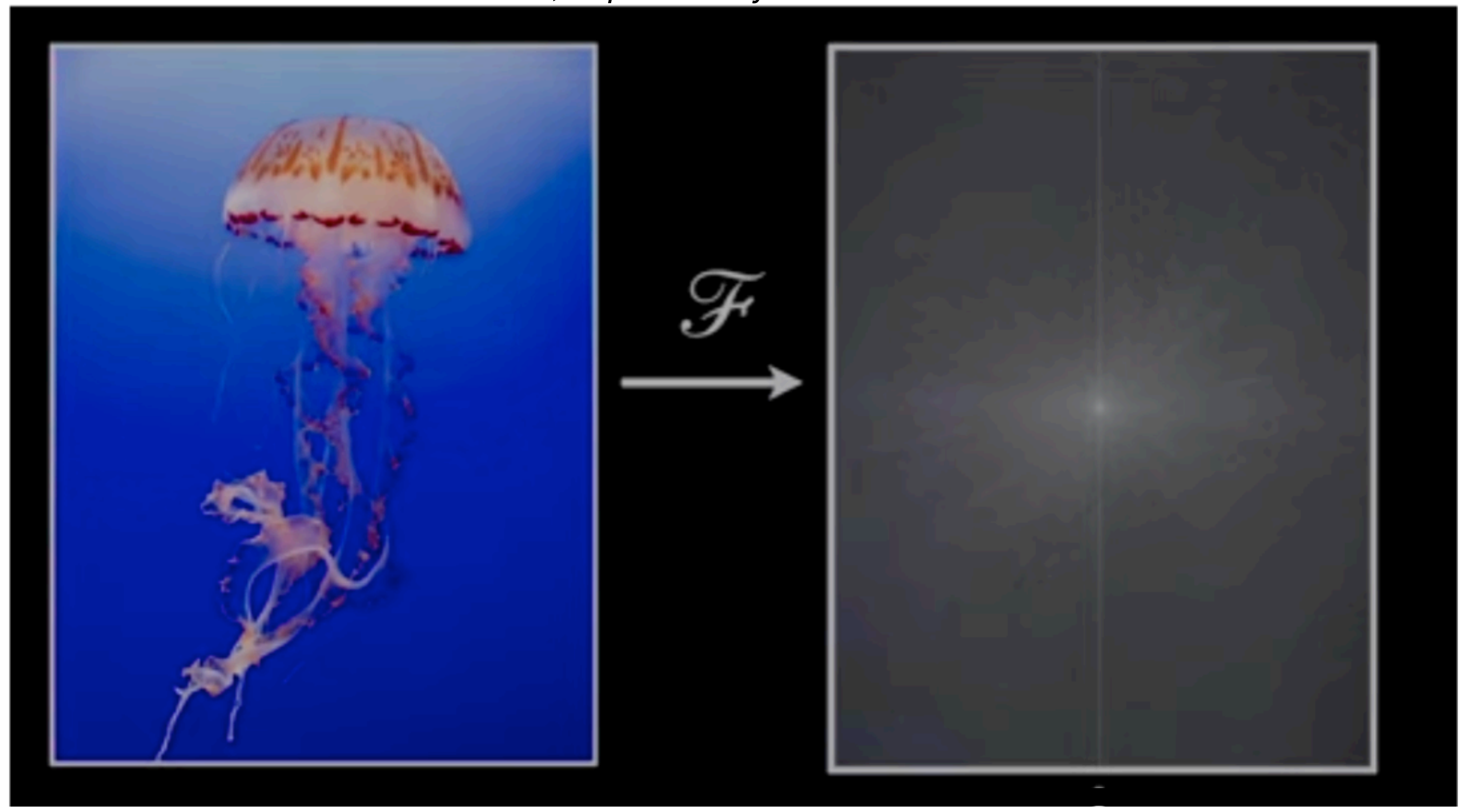


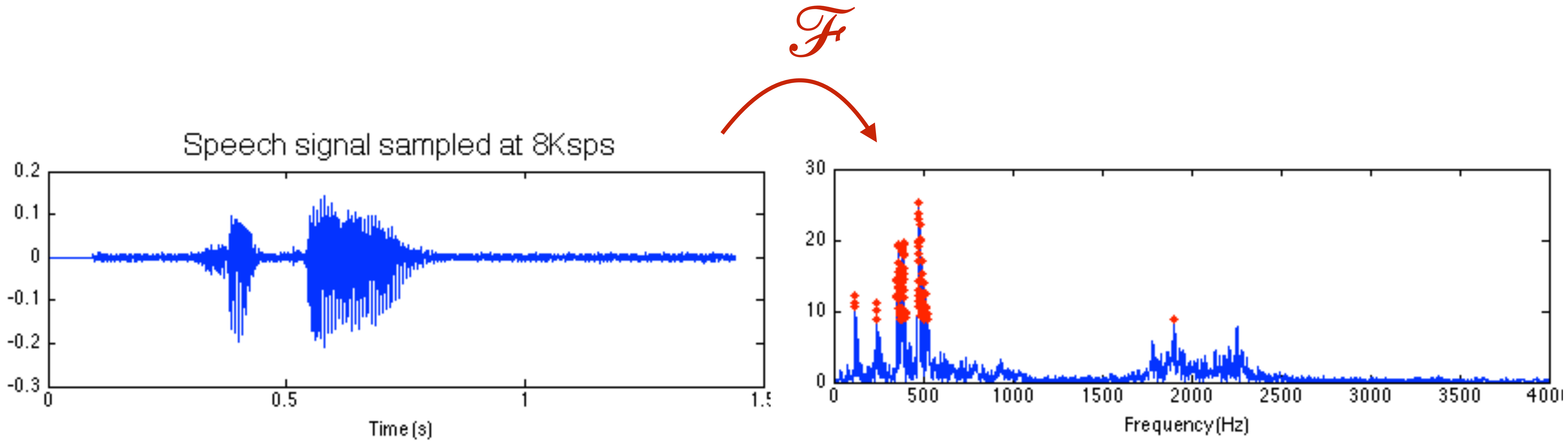


Real world signals are sparse in Fourier domain
(or some other such universal domain, such as Wavelet)

Motivating Compressive Sampling

Steve Brunton, https://www.youtube.com/watch?v=hmBTACBGWJs





Real world signals are sparse in Fourier domain
(or some other such universal domain, such as Wavelet)

$$x = \Psi s$$

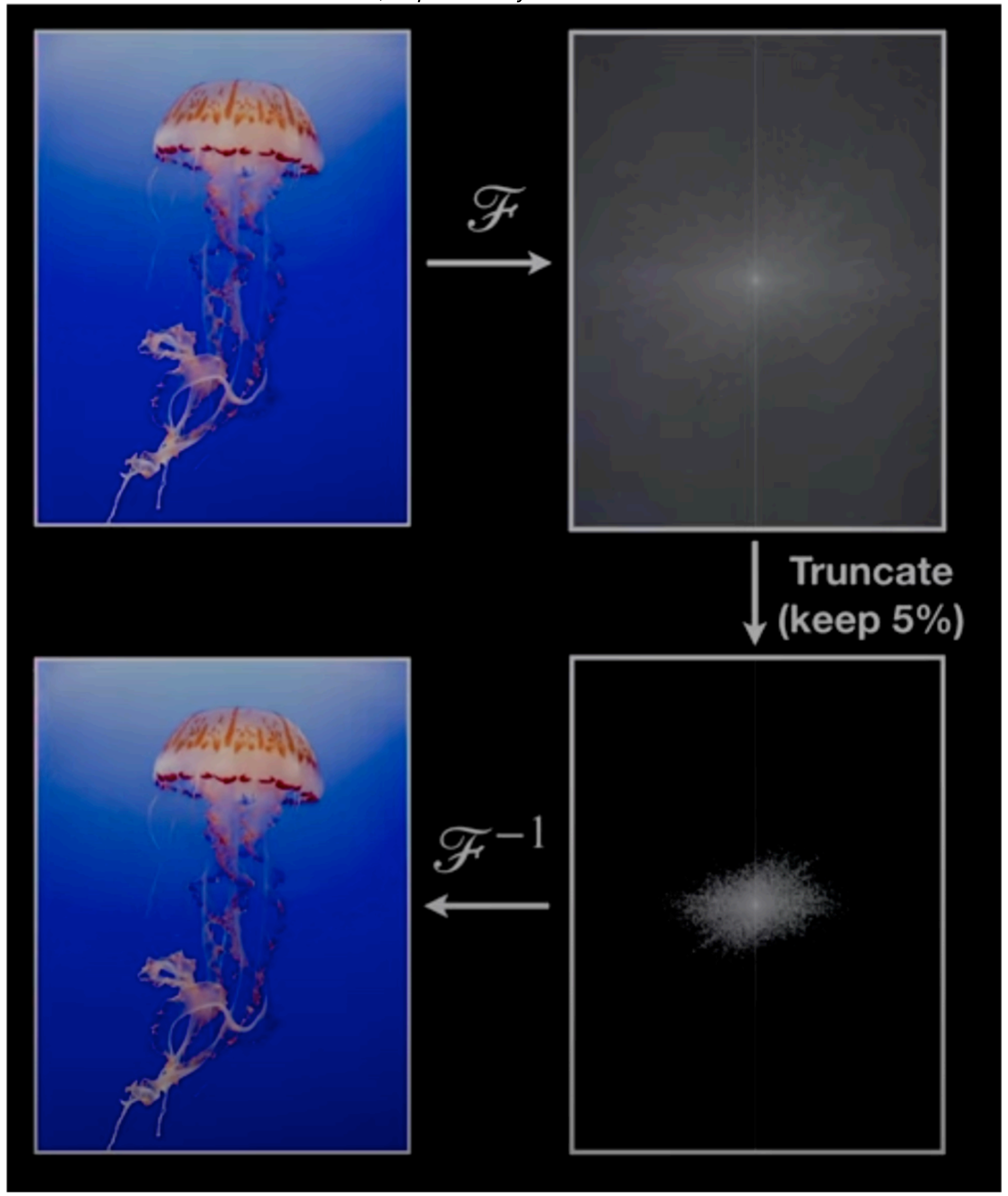
Fourier basis

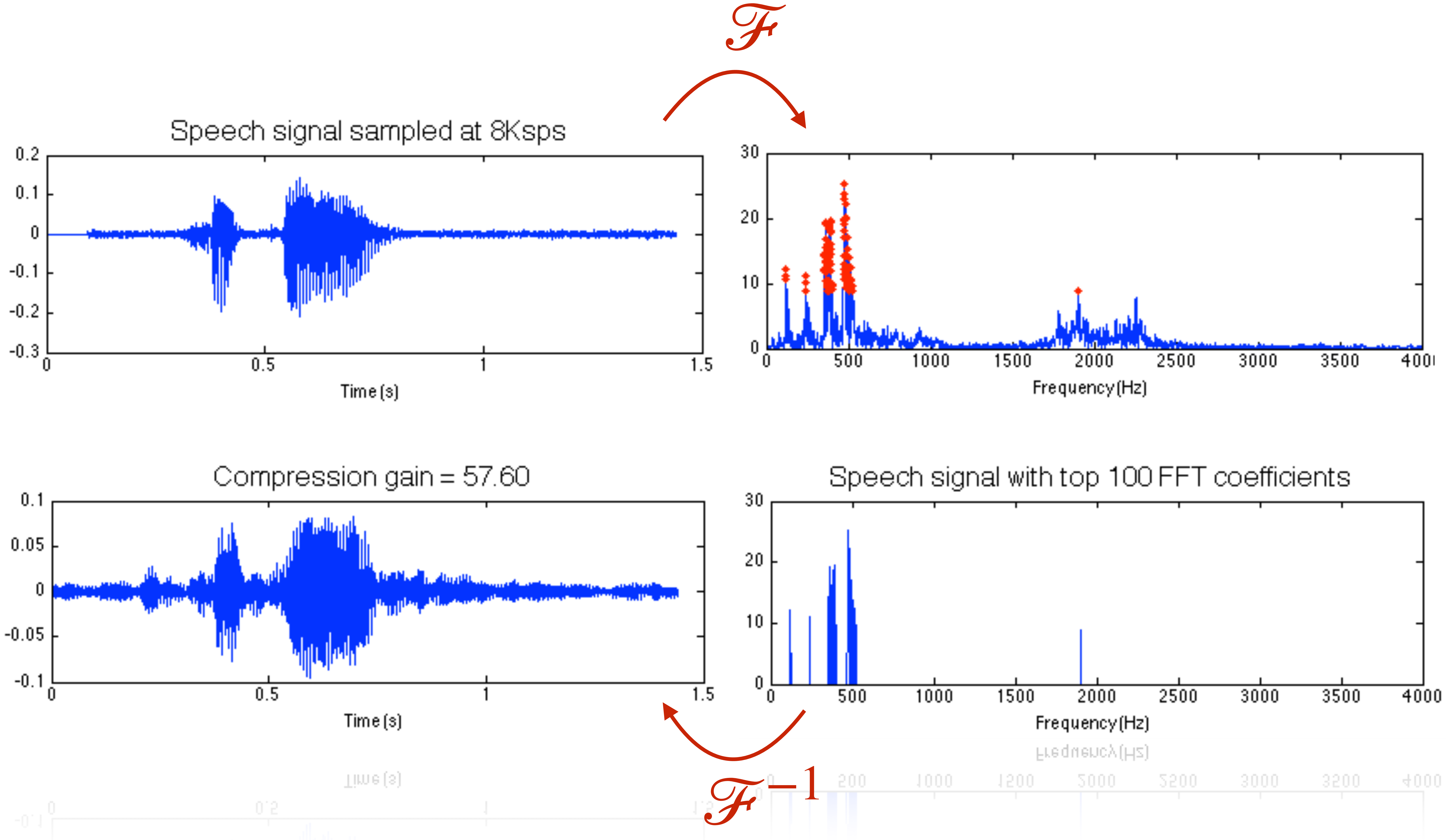
Dense signal
(e.g. image, audio)

Sparse

Motivating Compressive Sampling

Steve Brunton, https://www.youtube.com/watch?v=hmBTACBGWJs



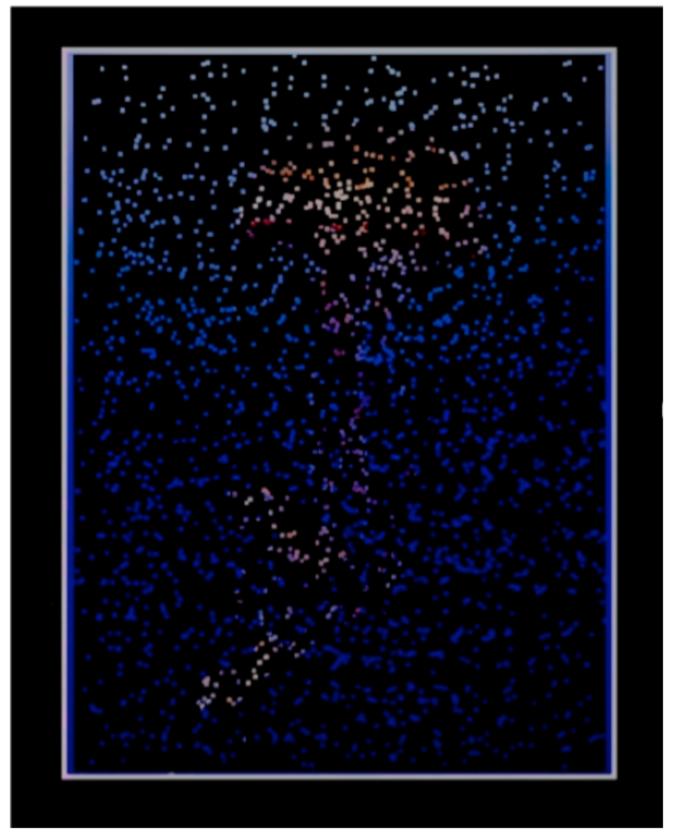


Loss compression methods exploit sparsity by retaining only top few coefficients that carry most of the signal energy.

Motivating Compressive Sampling

If we throw away most of the information during compression, why do we collect it to begin with? Can we not just starts with a massively downsampled signal?

Collect only
a few samples



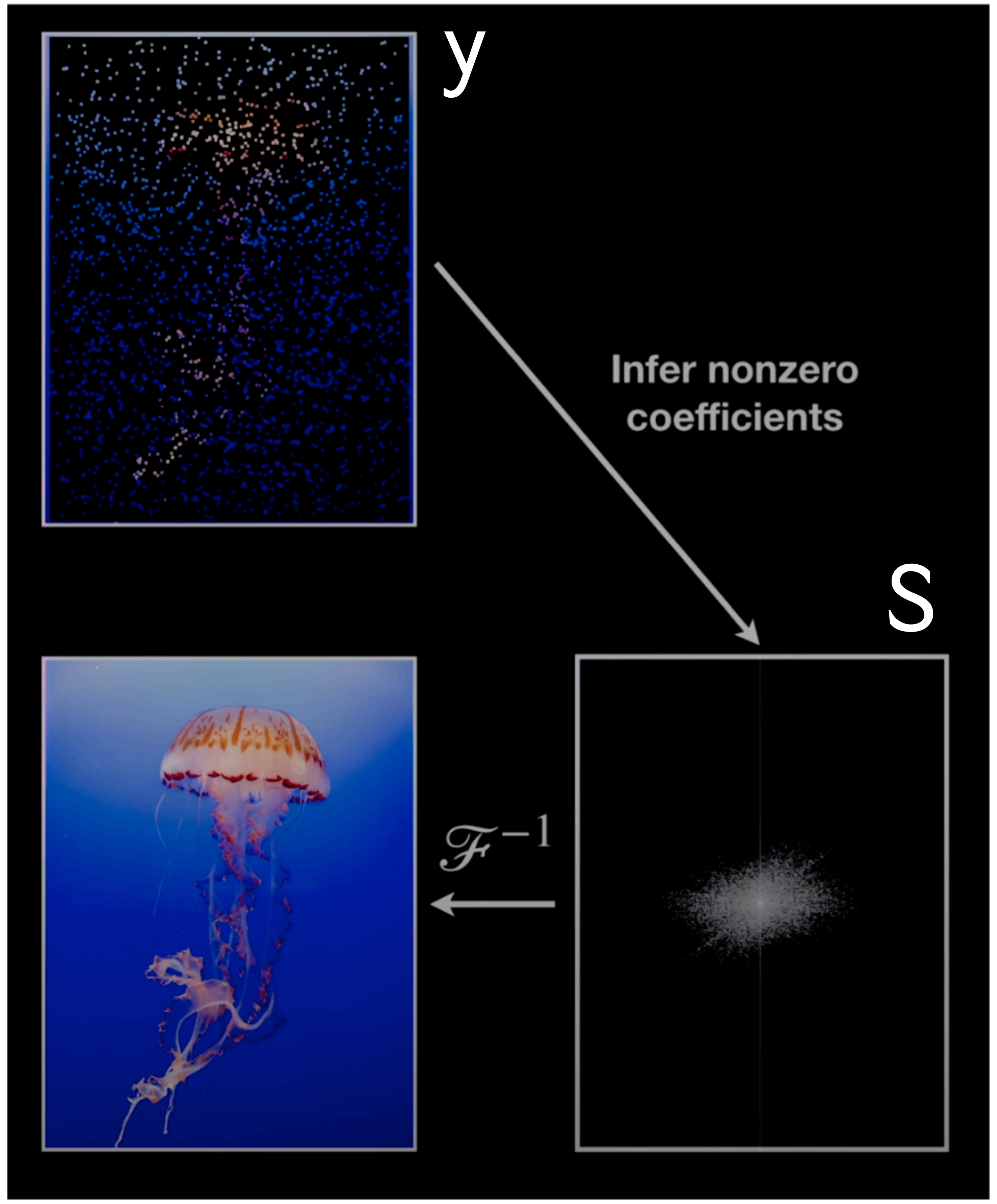
Steve Brunton, https://www.youtube.com/watch?v=hmBTACBGWJs

Motivating Compressive Sampling

If we throw away most of the information during compression, why do we collect it to begin with? Can we not just starts with a massively downsampled signal?

Steve Brunton, https://www.youtube.com/watch?v=hmBTACBGWJs

Collect only a few samples



Fourier basis

Dense signal

$$x = \Psi s$$

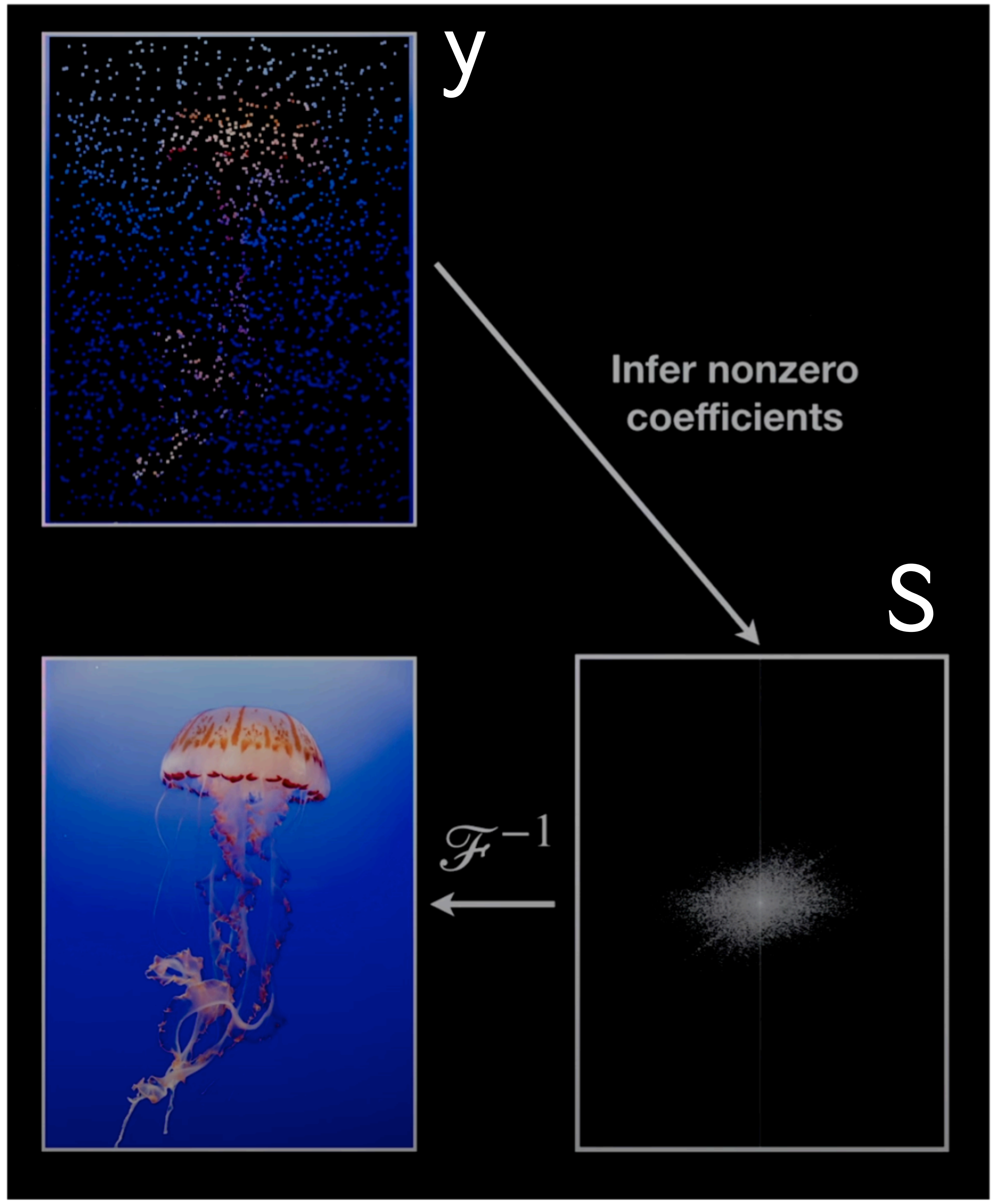
Sparse

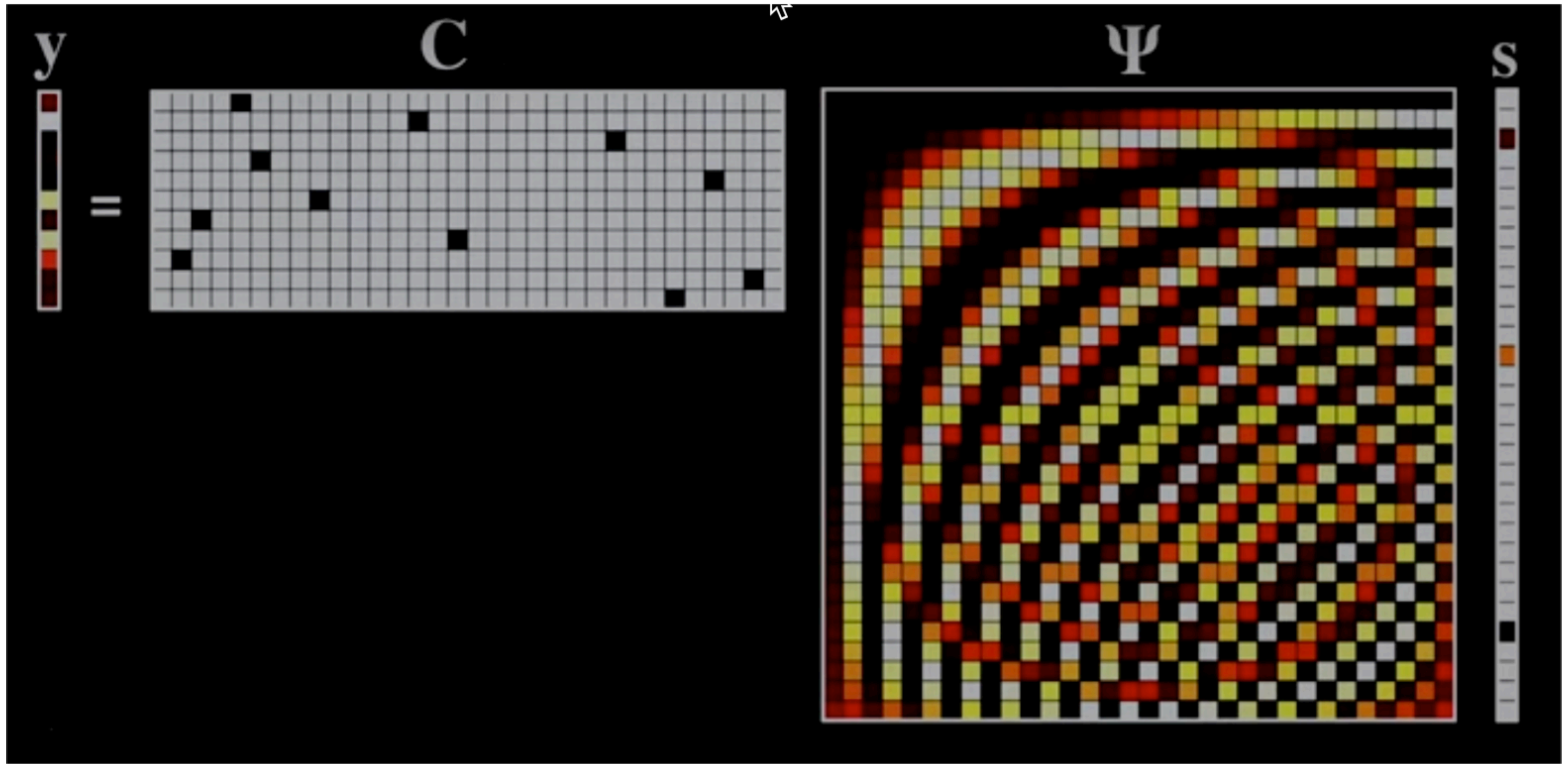
Measurement matrix

Measurements $|y| \ll |x|$

$$y = Cx$$
$$= C\Psi s$$
$$= \Theta s$$

Inferring s from y





$$y = Cx = C\Psi s = \Theta s$$

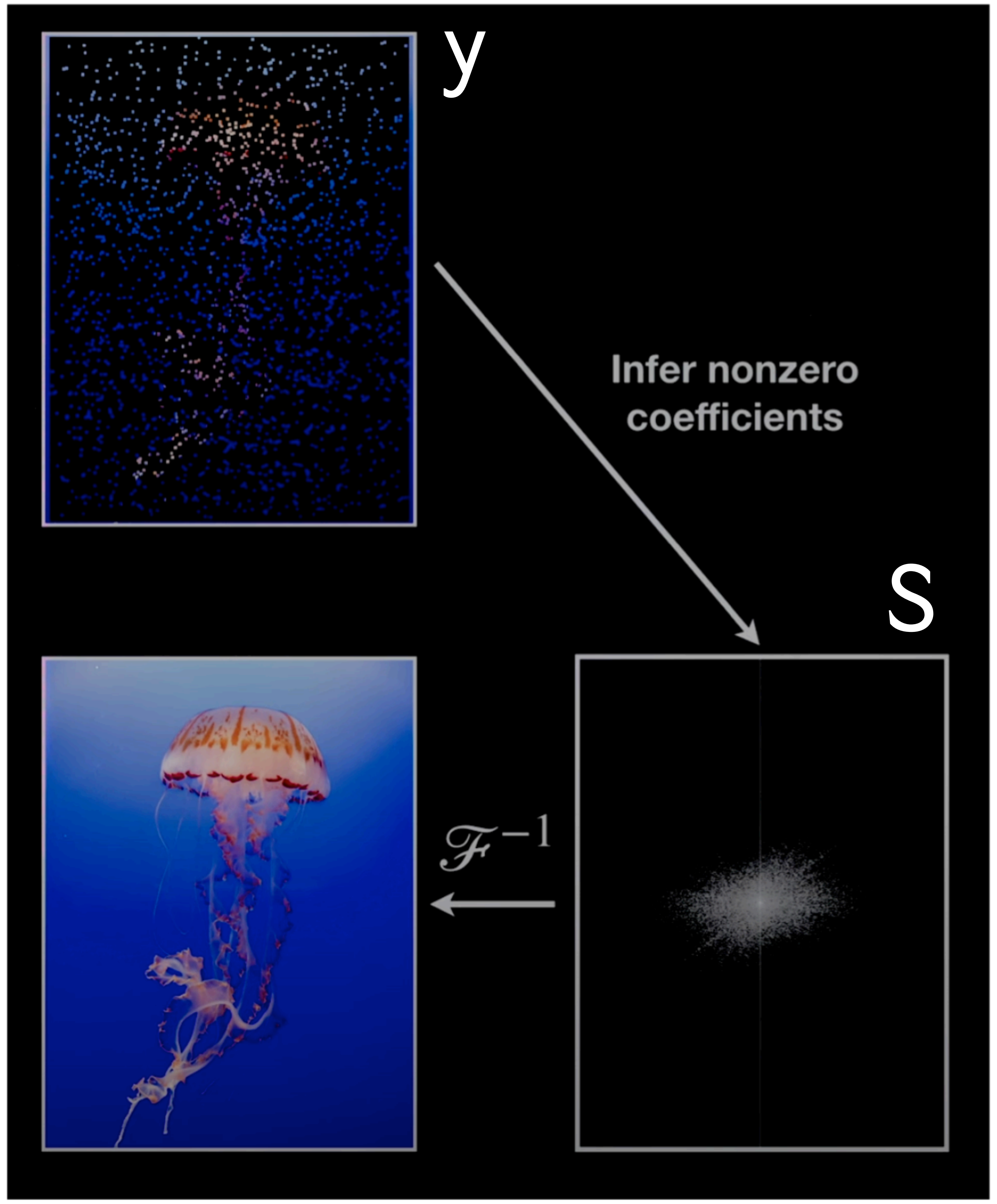
Undetermined Inverse Problem

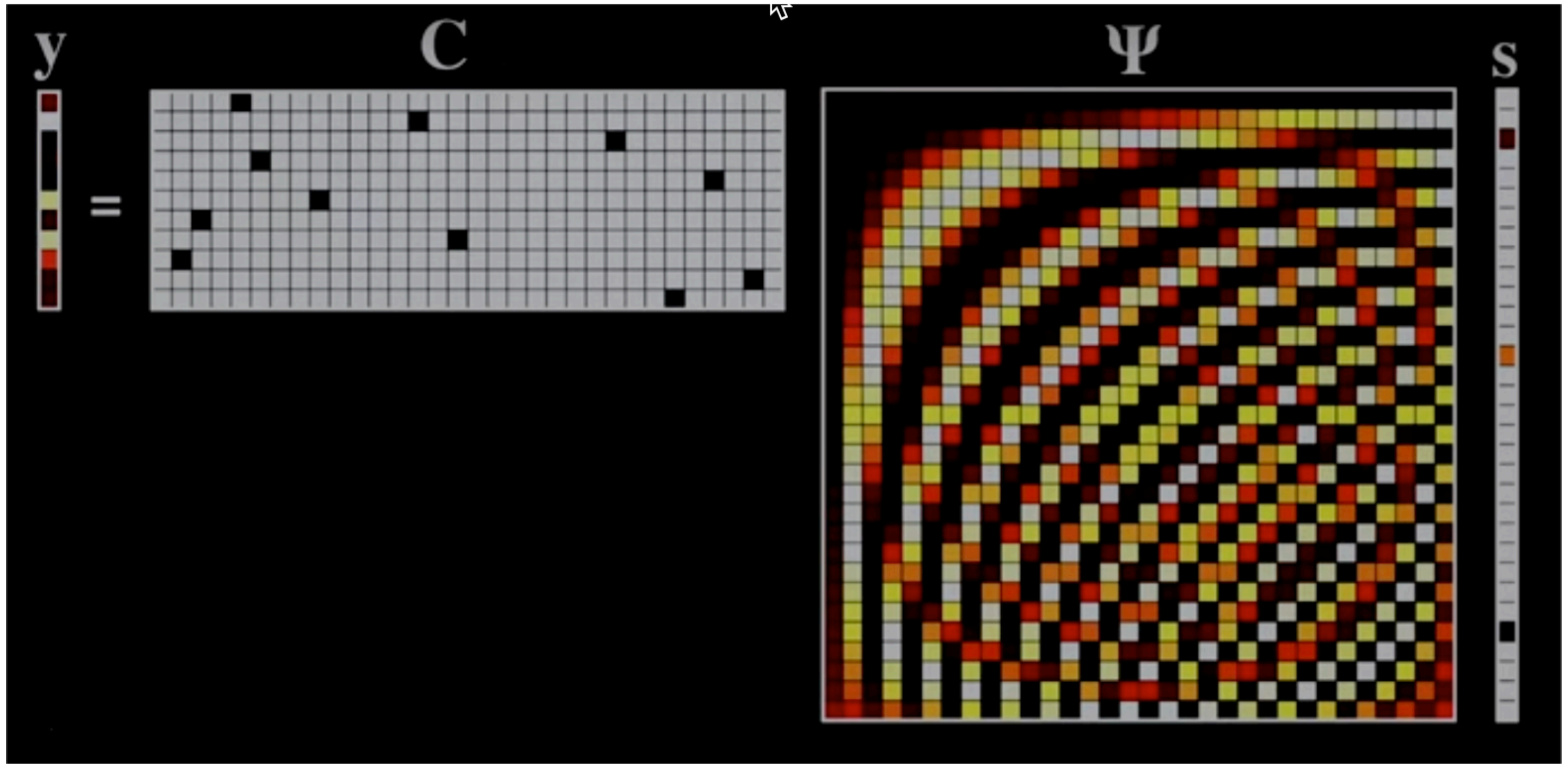
Solution for s given y and Θ is not unique.

Note: once we have **s** we can get **x** via inverse-FFT

Steve Brunton, https://www.youtube.com/watch?v=hmBTACBGWJs

Adding the Sparsity Requirement to Infer **s**





$$y = Cx = C\Psi s = \Theta s$$

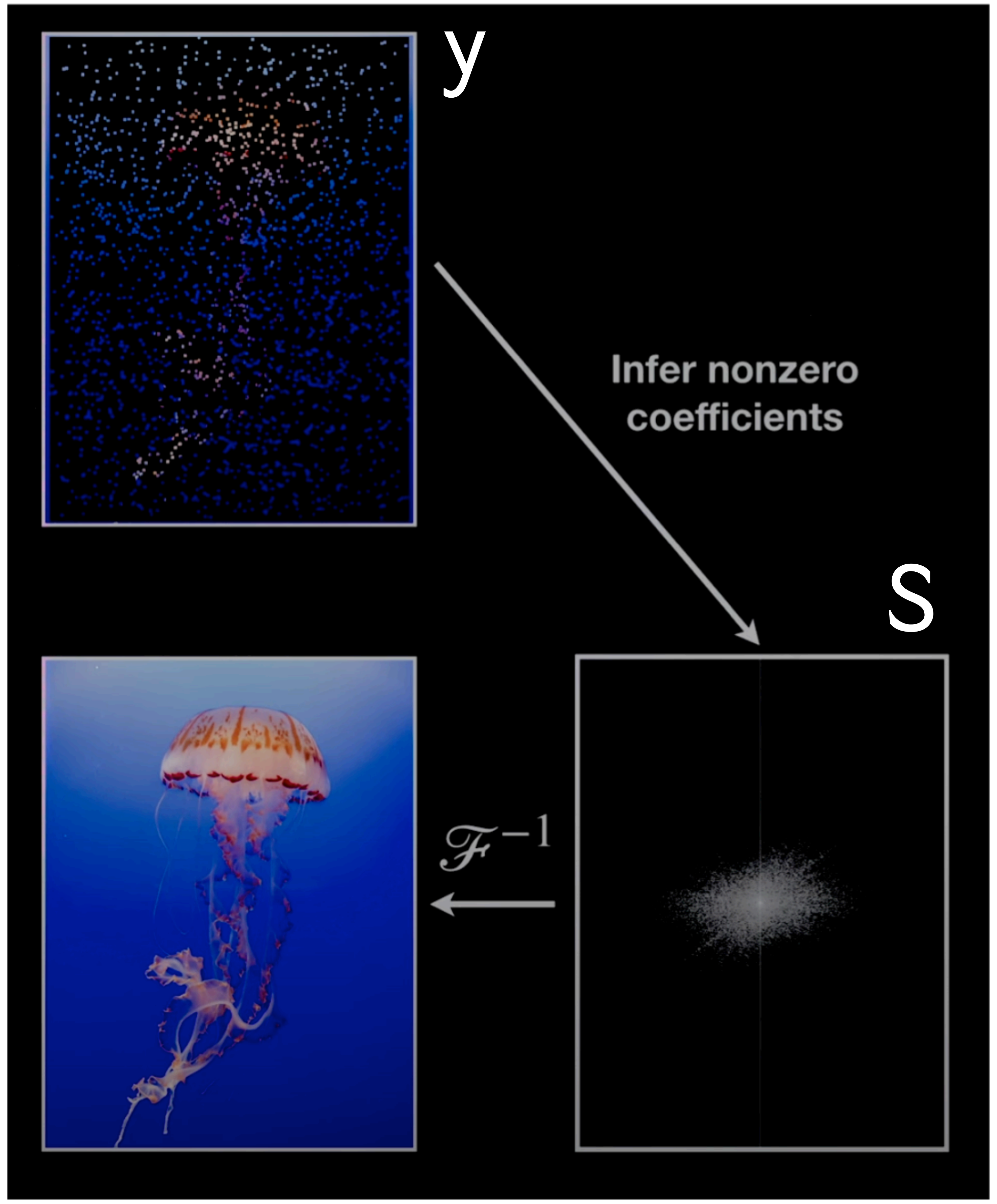
$$\hat{s} = \text{argmin}_{s'} ||C\Psi s' - y||_2 + \lambda ||s'||_0$$

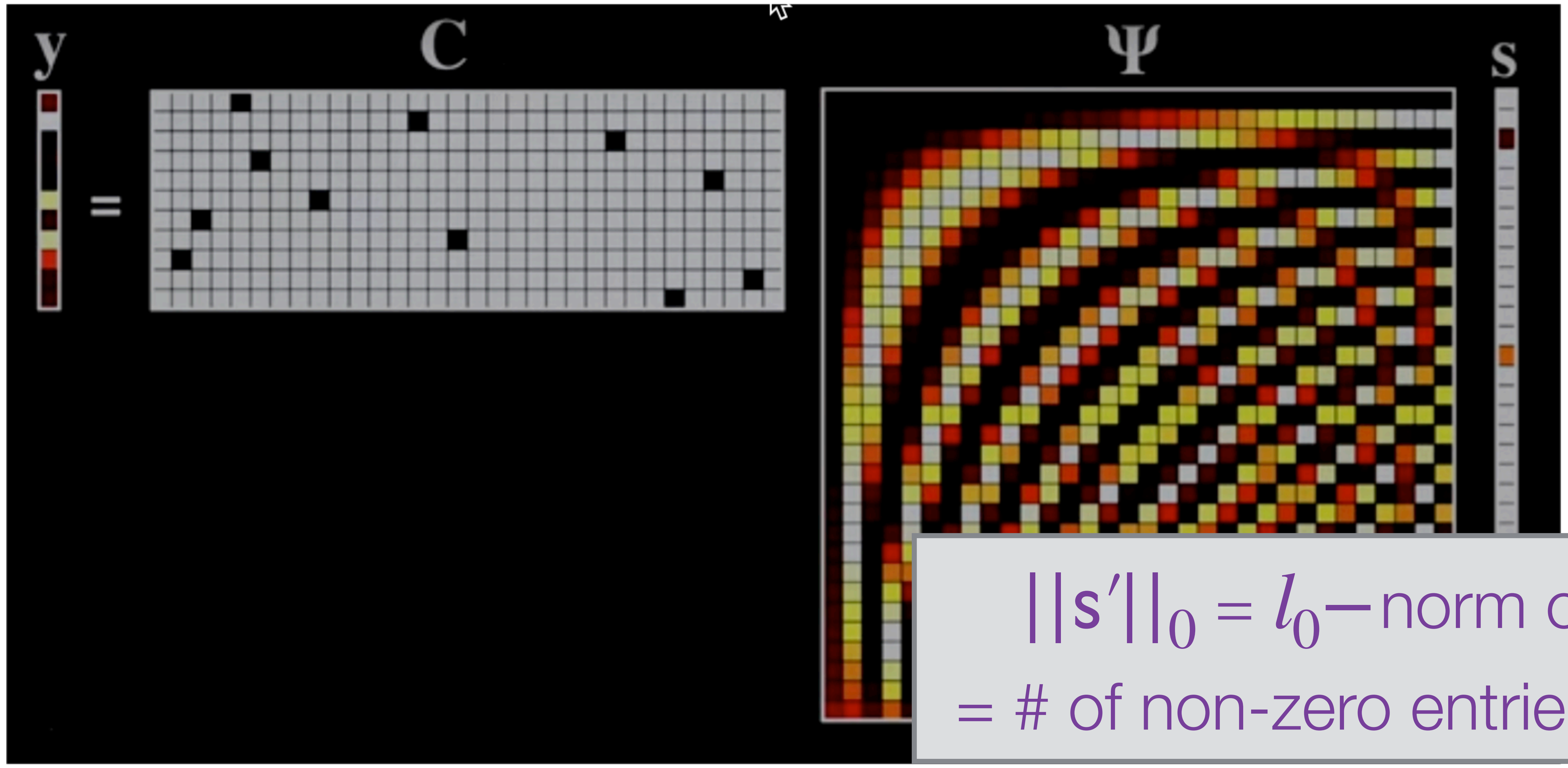
alternatively:

$$\hat{s} = \text{argmin}_{s'} ||s'||_0 \text{ , s.t. } ||C\Psi s' - y||_2 < \epsilon$$

Steve Brunton, https://www.youtube.com/watch?v=hmBTACBGWJs

Adding the Sparsity Requirement to Infer **s**





$||\mathbf{s}'||_0 = l_0$–norm of $\mathbf{s}'$

= # of non-zero entries in $\mathbf{s}'$

$$\mathbf{y} = \mathbf{C}\mathbf{x} = \mathbf{C}\Psi\mathbf{s} = \Theta\mathbf{s}$$

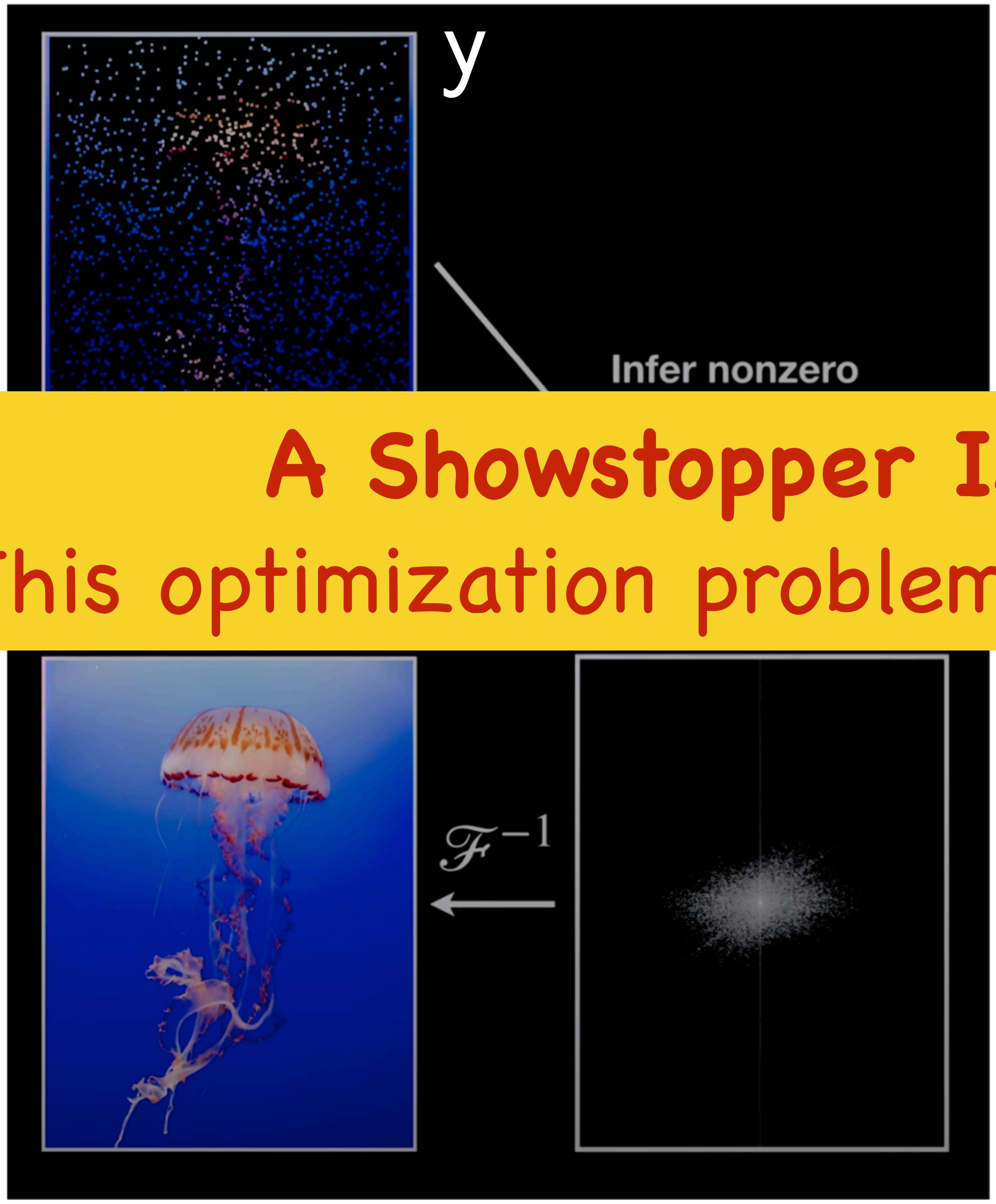
$$\hat{\mathbf{s}} = \text{argmin}_{s'} ||\mathbf{C}\Psi\mathbf{s}' - \mathbf{y}||_2 + \lambda||\mathbf{s}'||_0$$

alternatively:

$$\hat{\mathbf{s}} = \text{argmin}_{s'} ||\mathbf{s}'||_0 \text{ , s.t. } ||\mathbf{C}\Psi\mathbf{s}' - \mathbf{y}||_2 < \epsilon$$

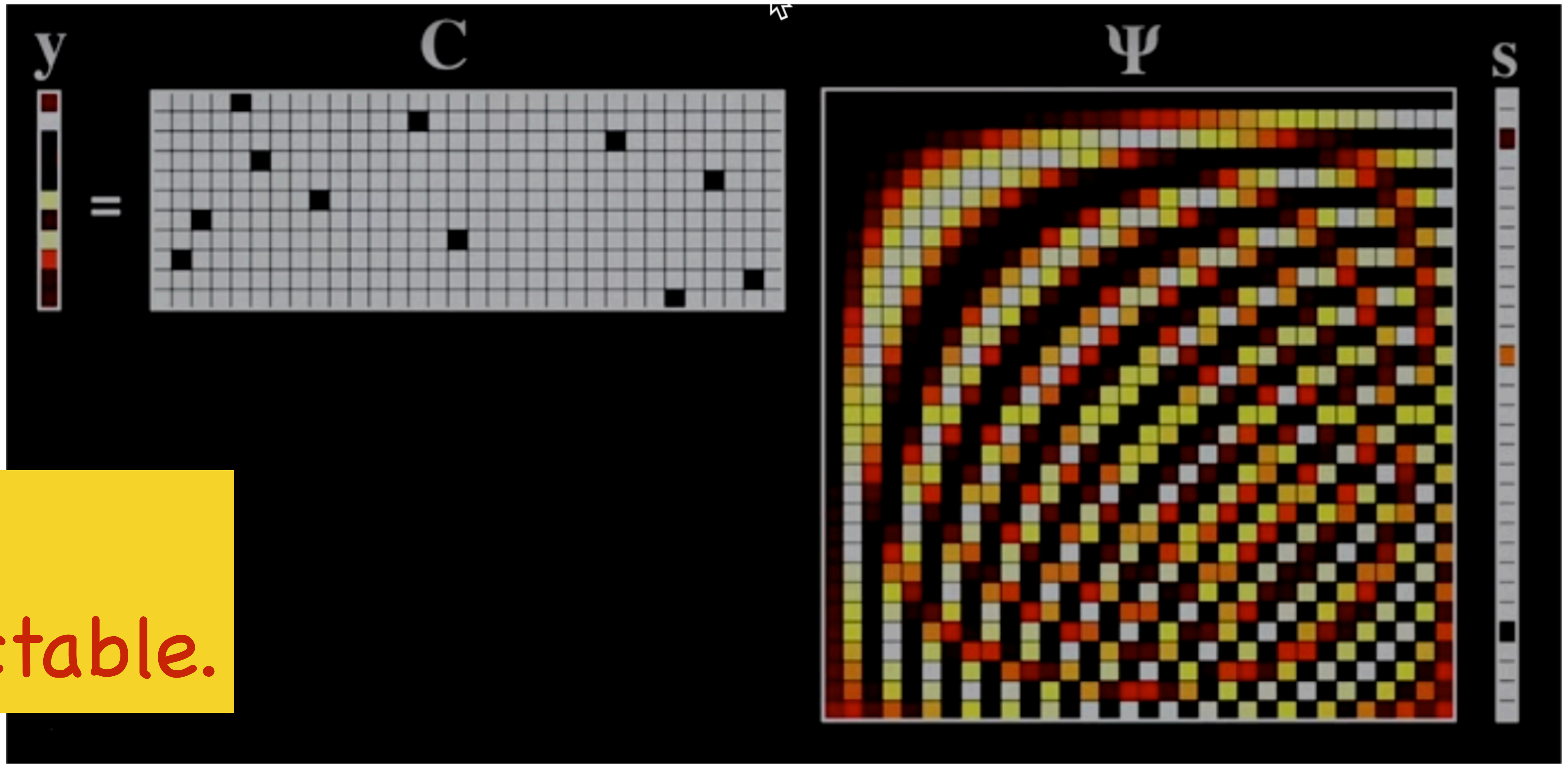
Steve Brunton, https://www.youtube.com/watch?v=hmBTACBGWJs

Adding the Sparsity Requirement to Infer **s**



A Showstopper Issue !!!

This optimization problem is intractable.



$$y = Cx = C\Psi s = \Theta s$$

$$\hat{s} = \text{argmin}_{s'} ||C\Psi s' - y||_2 + \lambda ||s'||_0$$

alternatively:

$$\hat{s} = \text{argmin}_{s'} ||s'||_0 \text{ , s.t. } ||C\Psi s' - y||_2 < \epsilon$$

Steve Brunton, https://www.youtube.com/watch?v=hmBTACBGWJs

A Big Applied Math Breakthrough to the Rescue

(~ 2004-2005 @ CalTech, Rice, UCLA)

$\hat{s} = \text{argmin}_{s'} ||C\Psi s' - y||_2 + \lambda ||s'||_0$

alternatively:

$\hat{s} = \text{argmin}_{s'} ||s'||_0$, s.t. $||C\Psi s' - y||_2 < \epsilon$

Computationally Intractable

$\hat{s} = \text{argmin}_{s'} ||C\Psi s' - y||_2 + \lambda ||s'||_1$

alternatively:

$\hat{s} = \text{argmin}_{s'} ||s'||_1$, s.t. $||C\Psi s' - y||_2 < \epsilon$

Computationally Efficient
(Convex Optimization)

Gives the exact solution for s with probability close to 1 if **certain conditions** are satisfied

A Big Applied Math Breakthrough to the Rescue

(~ 2004-2005 @ CalTech, Rice, UCLA)

$\hat{s} = \text{argmin}_{s'} ||C\Psi s' - y||_2 + \lambda||s'||_0$

alternatively:

$\hat{s} = \text{argmin}_{s'} ||s'||_0$, s.t. $||C\Psi s' - y||_2 < \epsilon$

Computationally Intractable

→

$\hat{s} = \text{argmin}_{s'} ||C\Psi s' - y||_2 + \lambda||s'||_1$

alternatively:

$\hat{s} = \text{argmin}_{s'} ||s'||_1$, s.t. $||C\Psi s' - y||_2 < \epsilon$

Computationally Efficient (Convex Optimization)

- C should be incoherent w.r.t. Ψ (i.e. rows of C should not be too parallel to columns of Ψ)
- # of measurements $p \sim O(k\log(n/k))$ where $k = ||s||_0$ and $n = |s| = |x|$

Gives the exact solution for s with probability close to 1 if **certain conditions** are satisfied

A Big Applied Math Breakthrough to the Rescue

(~ 2004-2005 @ CalTech, Rice, UCLA)

$\hat{s} = \text{argmin}_{s'} ||C\Psi s' - y||_2 + \lambda ||s'||_0$

alternatively:

$\hat{s} = \text{argmin}_{s'} ||s'||_0$, s.t. $||C\Psi s' - y||_2 < \epsilon$

Computationally Intractable

→

$\hat{s} = \text{argmin}_{s'} ||C\Psi s' - y||_2 + \lambda ||s'||_1$

alternatively:

$\hat{s} = \text{argmin}_{s'} ||s'||_1$, s.t. $||C\Psi s' - y||_2 < \epsilon$

Computationally Efficient (Convex Optimization)

- C should be incoherent w.r.t. Ψ (i.e. rows of C should not be too parallel to columns of Ψ)
- # of measurements $p \sim O(k\log(n/k))$ where $k = ||s||_0$ and $n = |s| = |x|$

Restricted Isometry Property (RIP)

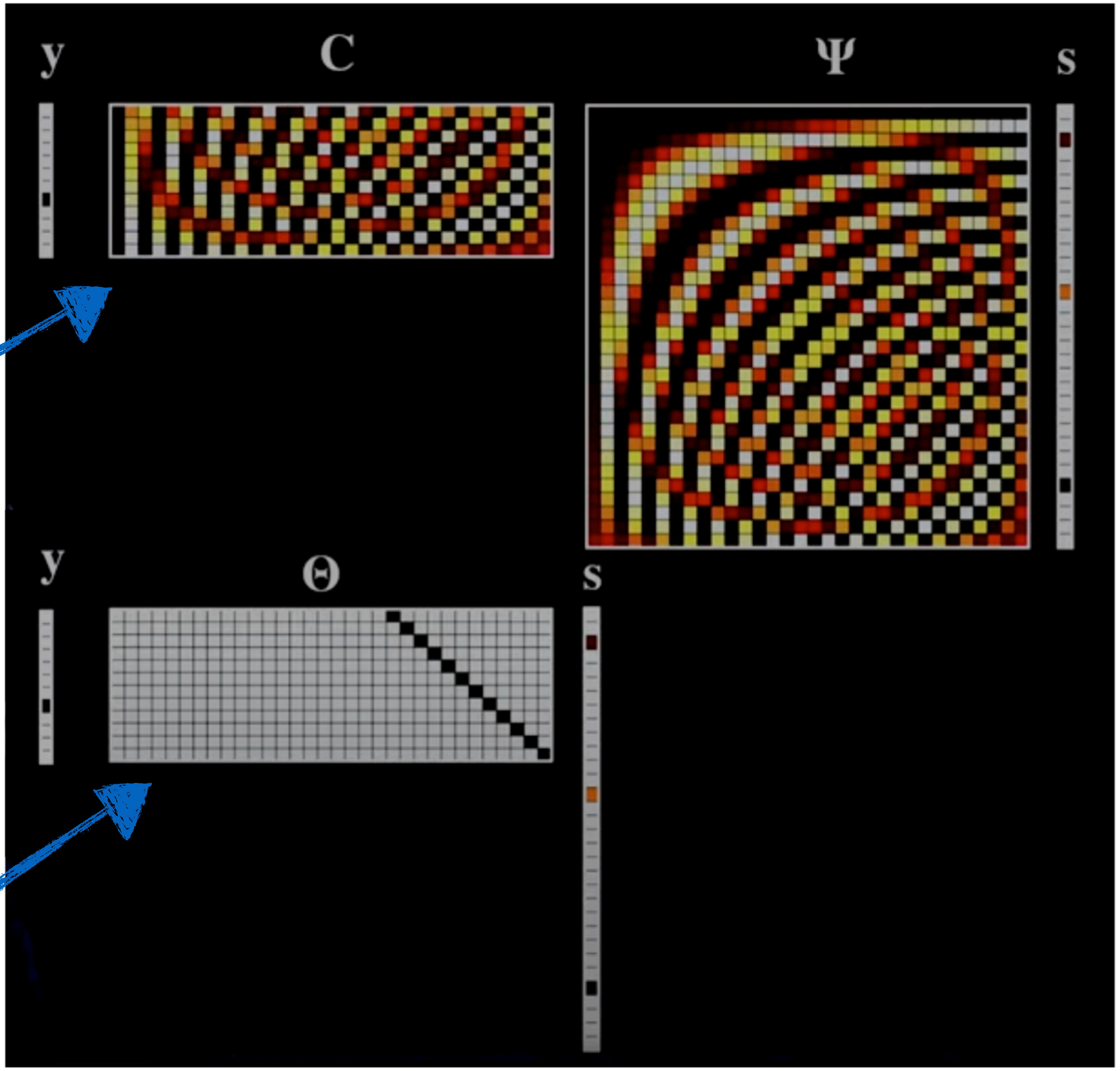
i.e. $C\Psi$ acts like a unitary matrix on sparse vector s

Gives the exact solution for s with probability close to 1 if **certain conditions** are satisfied

Example of Bad Measurement Matrix C

C has rows that are the same as a subset of columns of Ψ

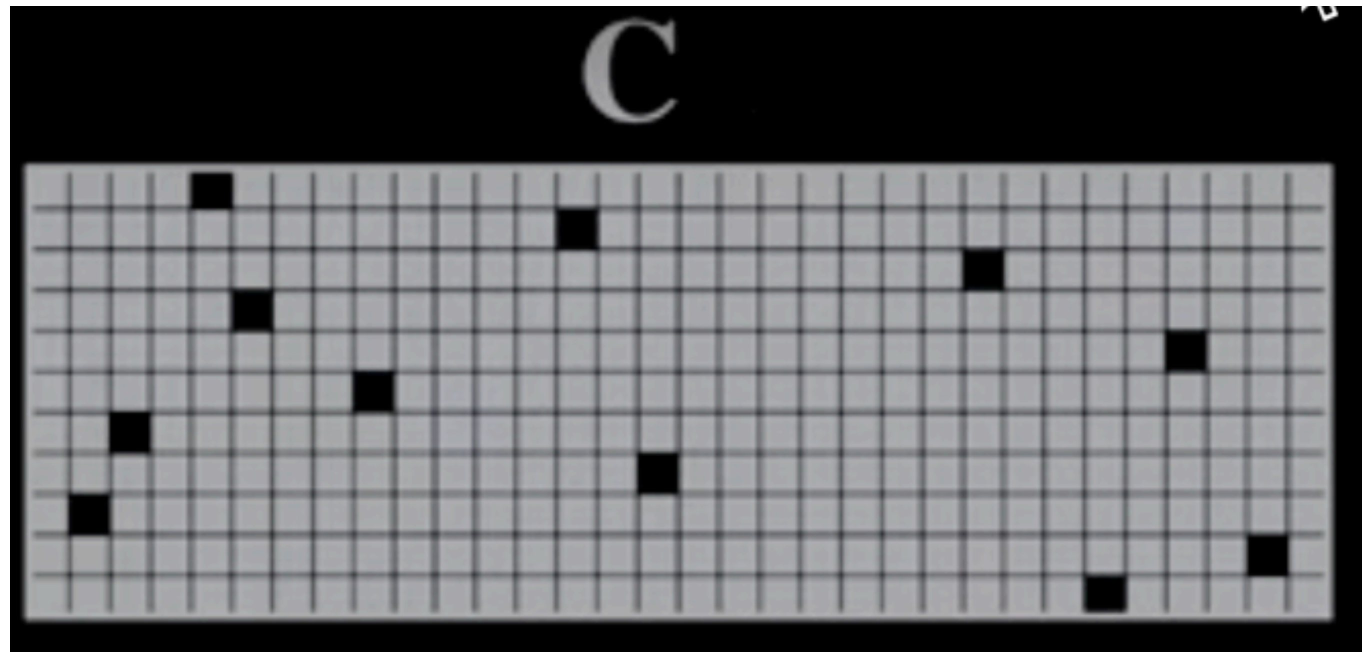
Θ only picks up information only from a part of s



Steve Brunton, https://www.youtube.com/watch?v=hmBTACBGWJs

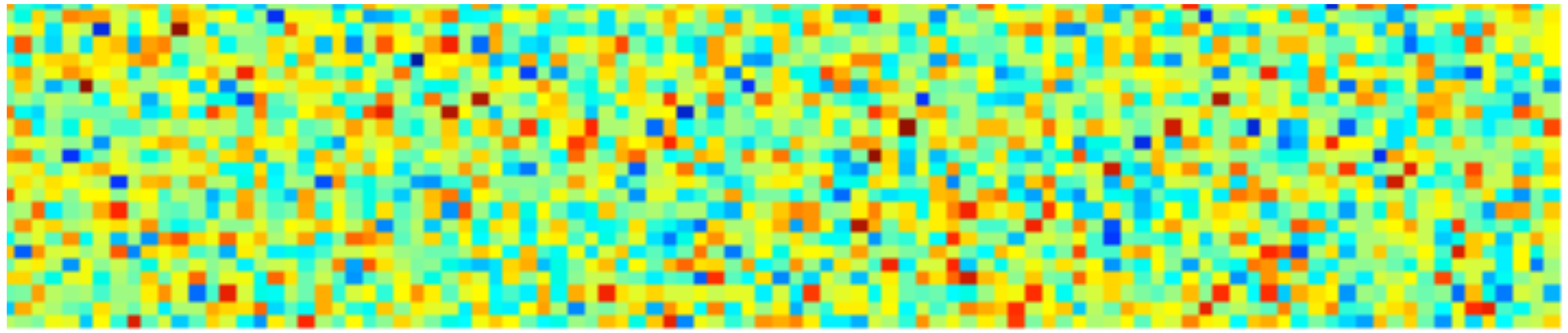
Measurement Matrix C in Practice

- **Random 0-1 mask matrix:** each element in y is a random sample picked from x
  - rows of C have exactly one 1 to indicate the sample picked by that measurement
  - must be causal in case of real-time sampling of a time-series



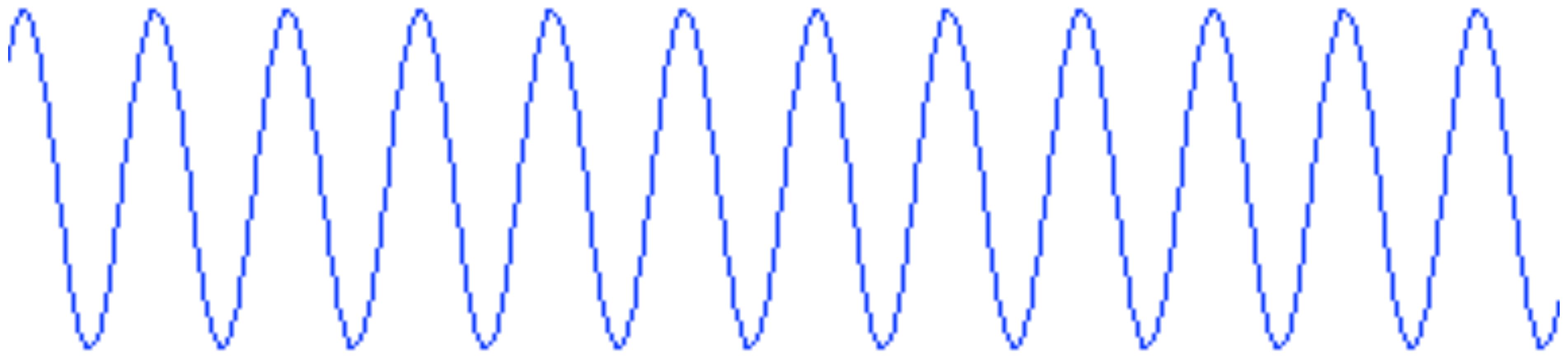
Steve Brunton, https://www.youtube.com/watch?v=hmBTACBGWJs

- **Random real-numbers:** each element in y is an incoherent random projection of x
  - commonly used: Gaussian, Bernoulli



Causal Random Sampling of a Time Series

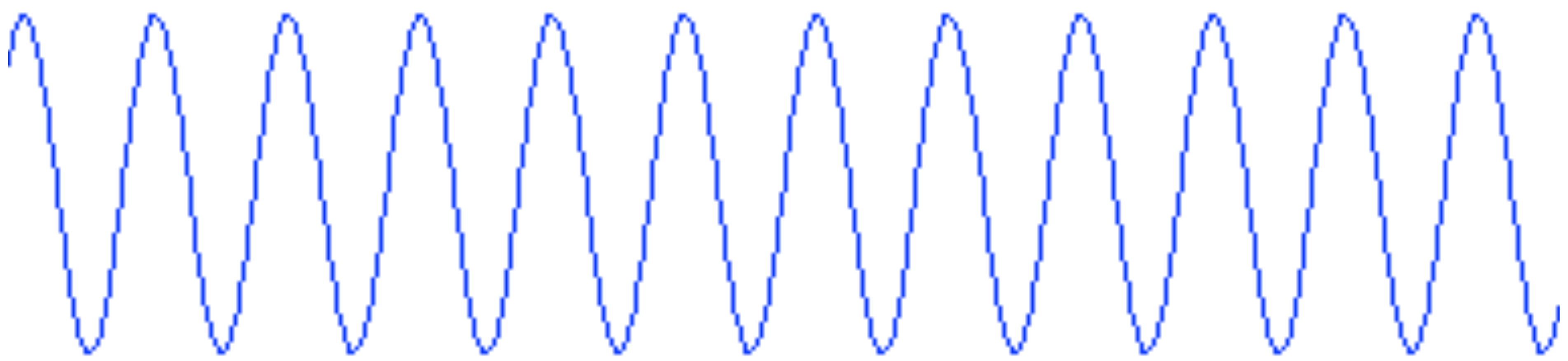
x



Signal to be acquired

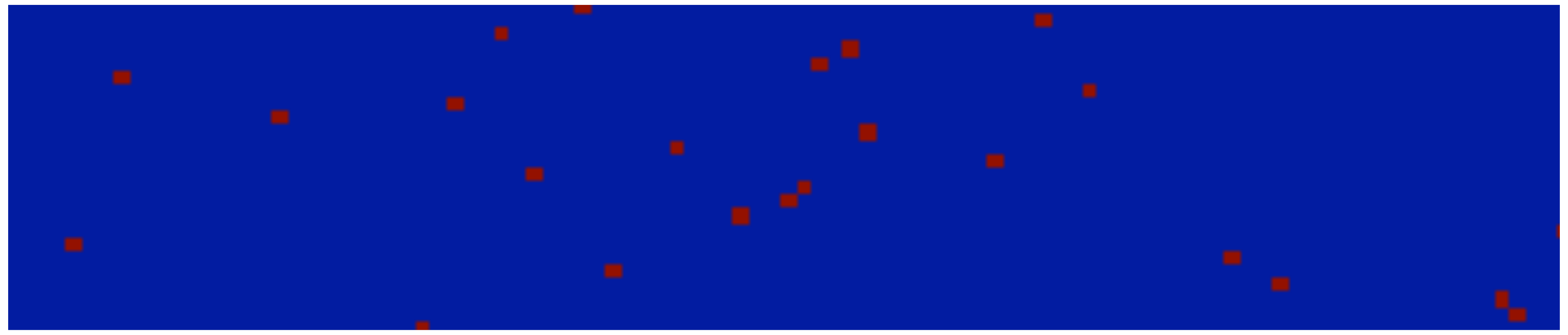
Causal Random Sampling of a Time Series

x



Signal to be acquired

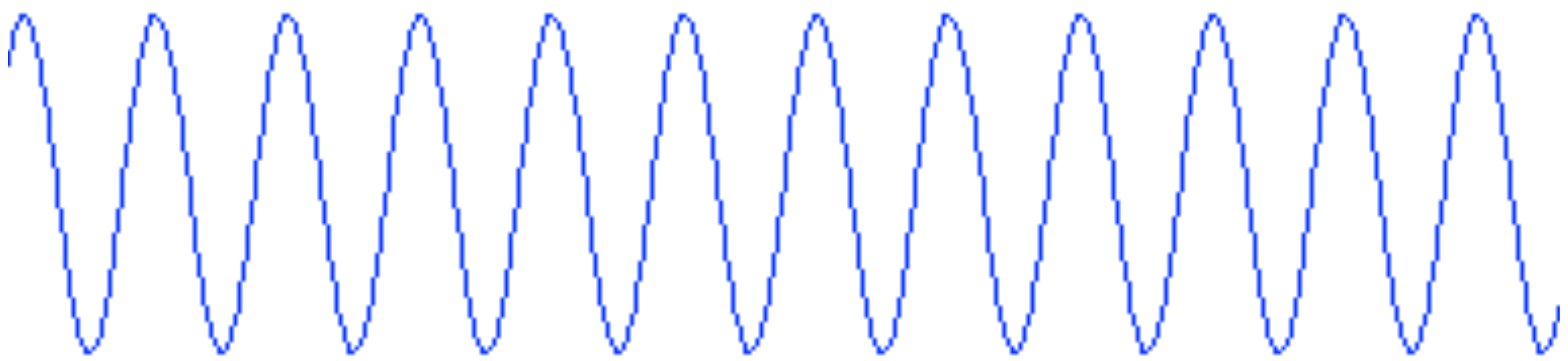
C



100 x 100 identity matrix with 75 rows uniformly randomly selected and thrown out

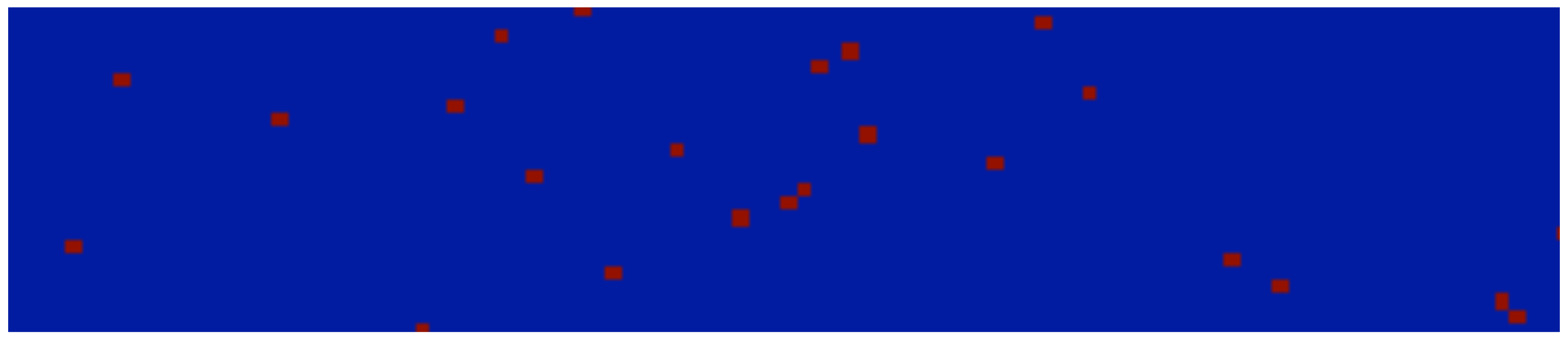
Causal Random Sampling of a Time Series

x



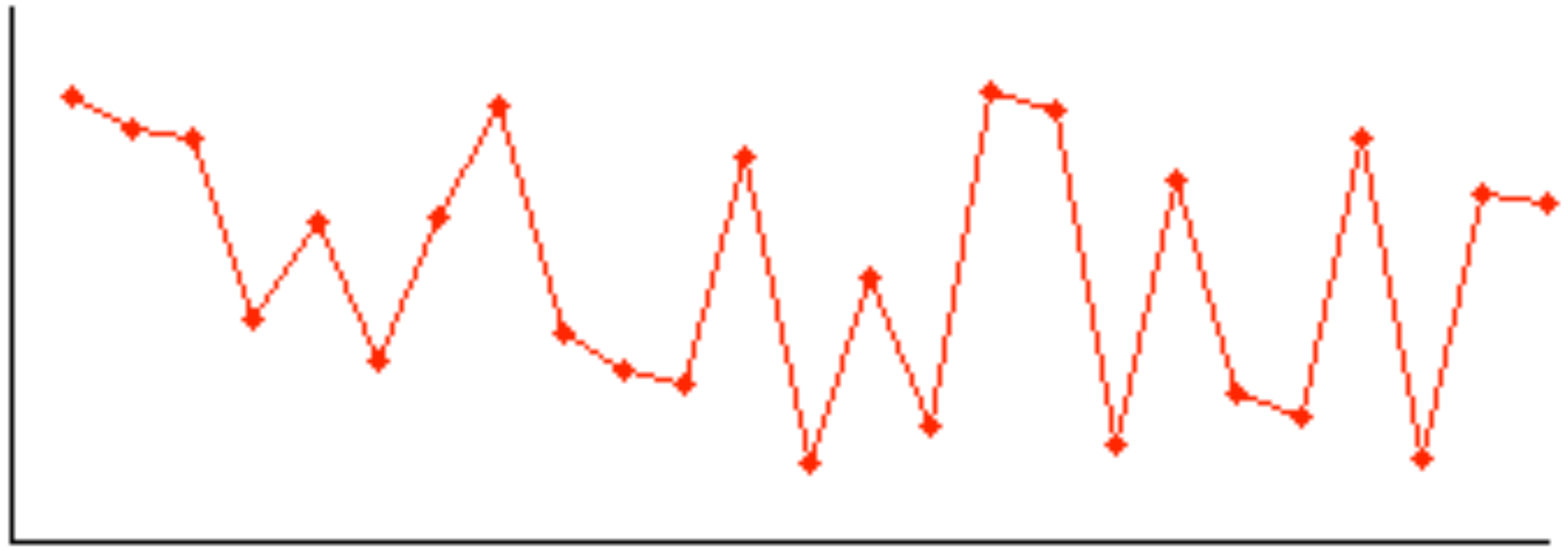
Signal to be acquired

C



100 x 100 identity matrix with 75 rows uniformly randomly selected and thrown out

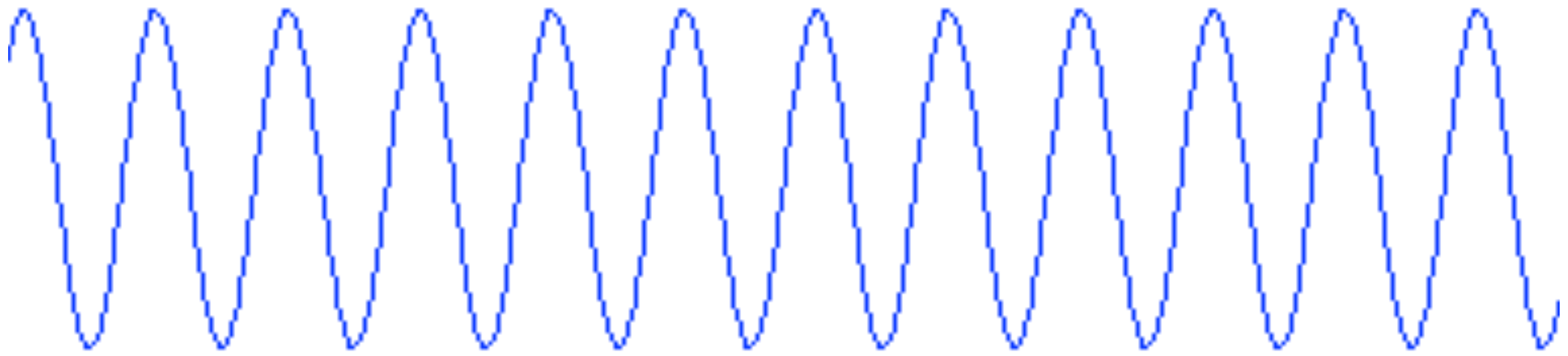
y



Measurements actually acquired

Taking Incoherent Projections of a Time Series - Gaussian

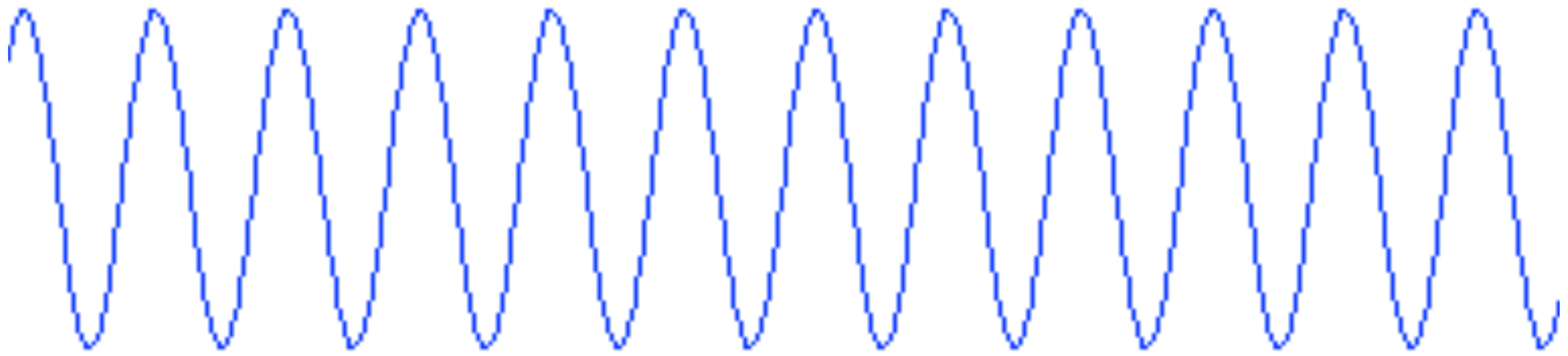
x



Signal to be acquired

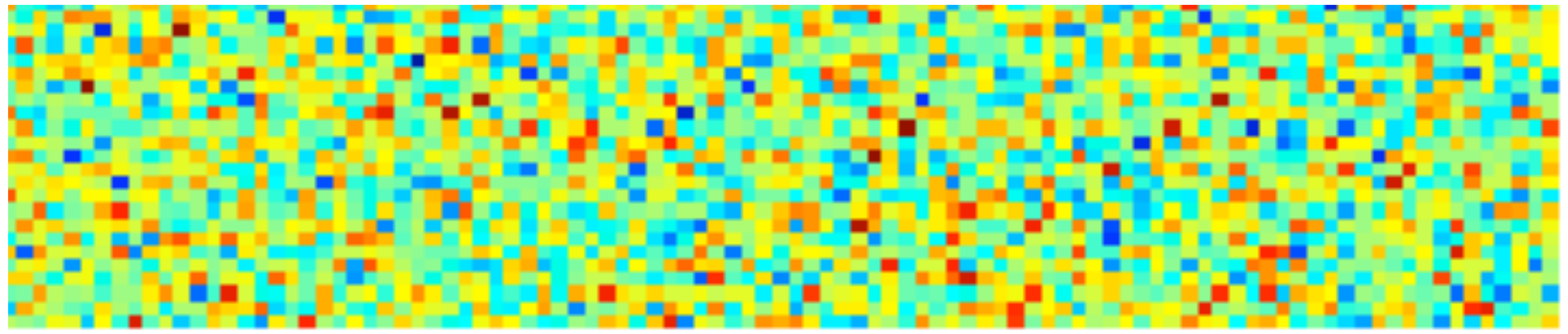
Taking Incoherent Projections of a Time Series - Gaussian

x



Signal to be acquired

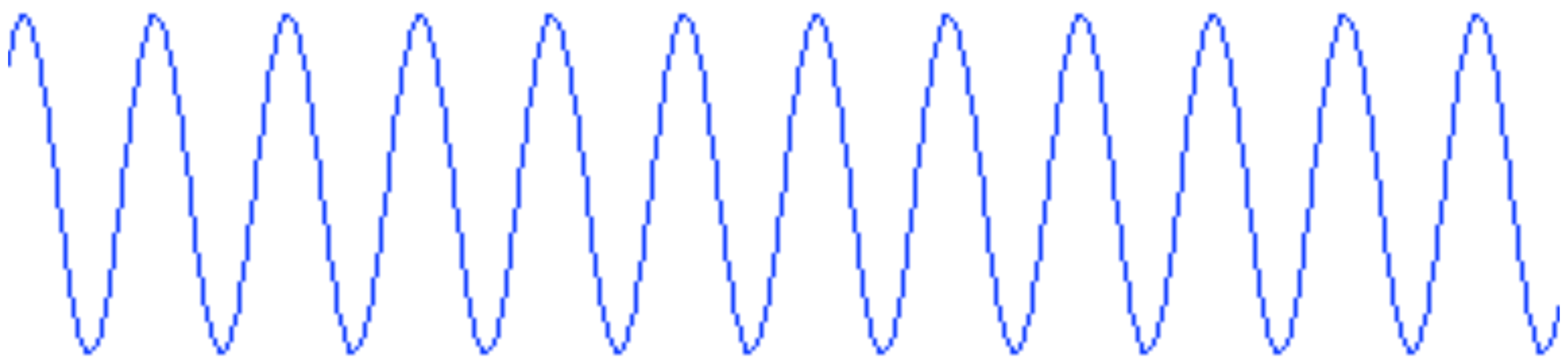
C



25 x 100 matrix of Gaussian $\mathcal{N}(0, \frac{1}{n})$ **distributed random numbers**

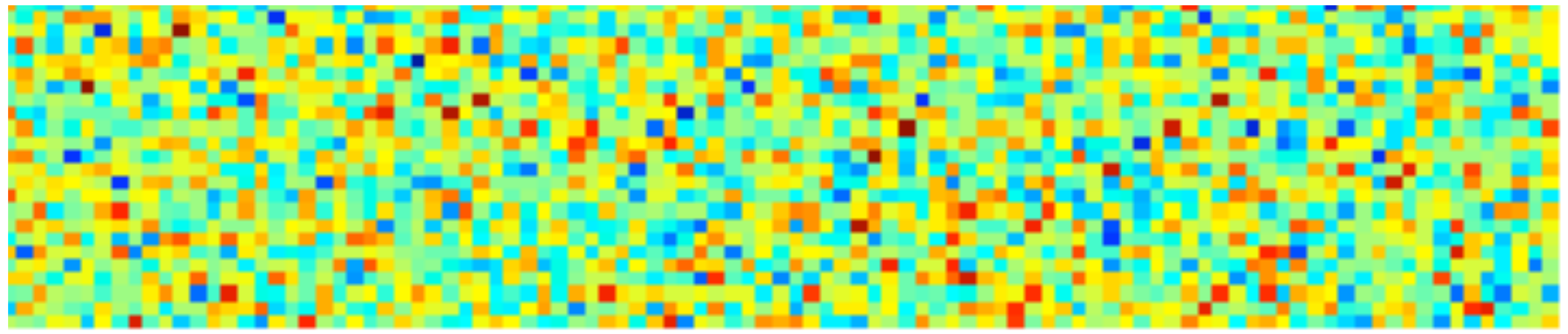
Taking Incoherent Projections of a Time Series - Gaussian

x



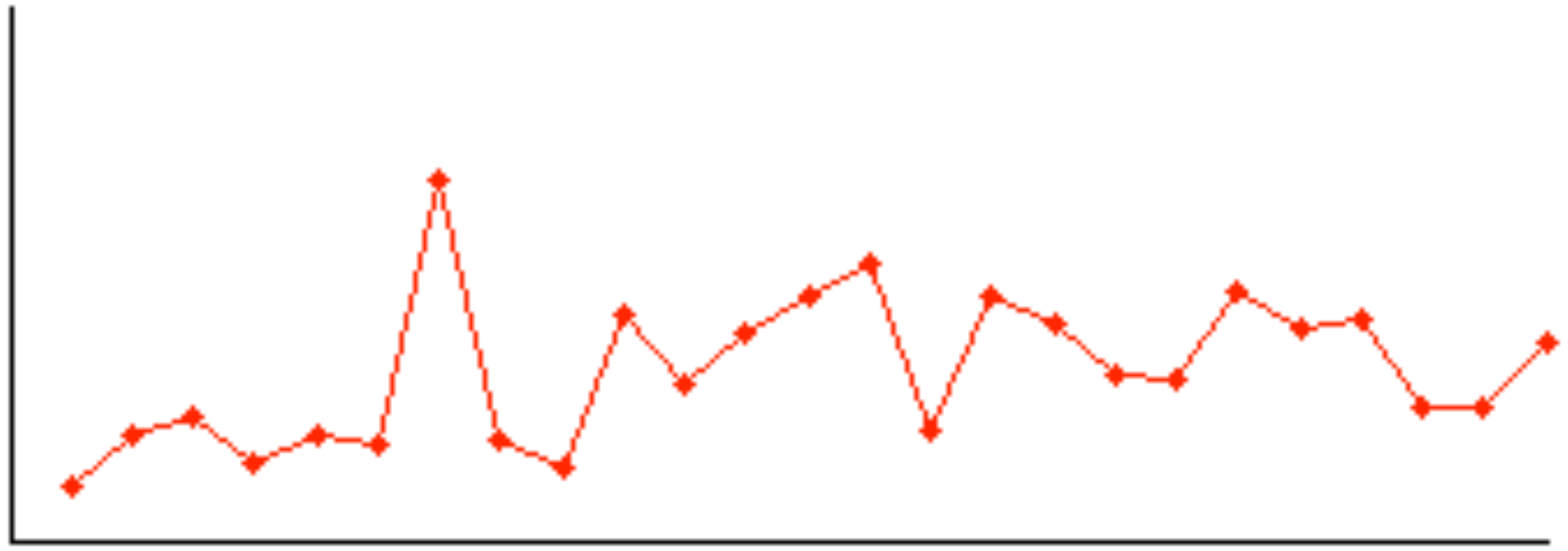
Signal to be acquired

C



25 x 100 matrix of Gaussian $\mathcal{N}(0,\frac{1}{n})$ distributed random numbers

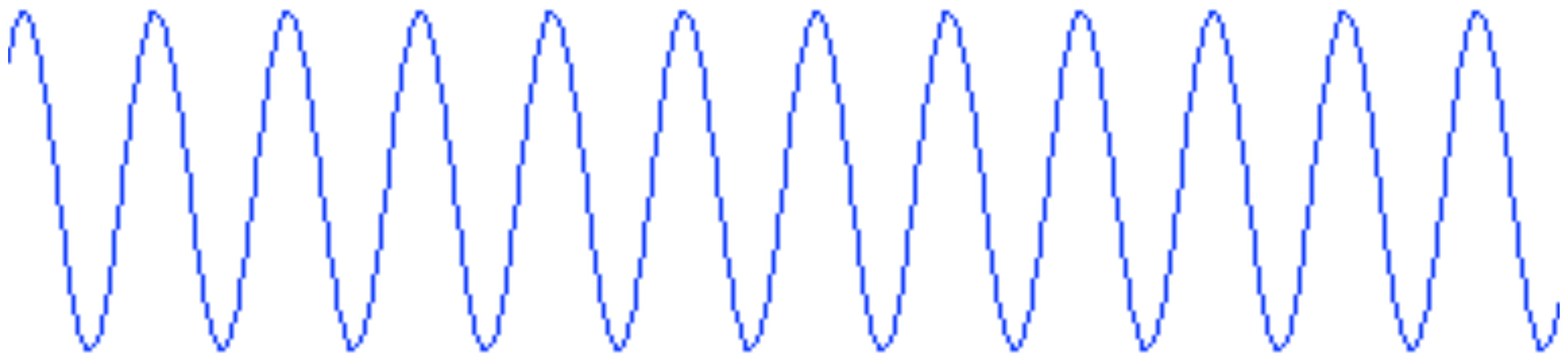
y



Measurements actually acquired

Taking Incoherent Projections of a Time Series - Bernoulli

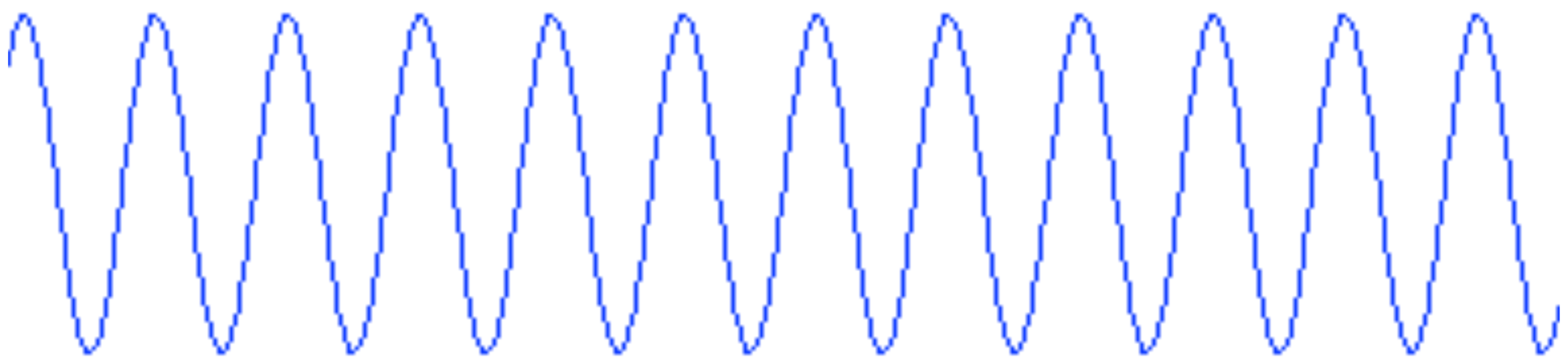
x



Signal to be acquired

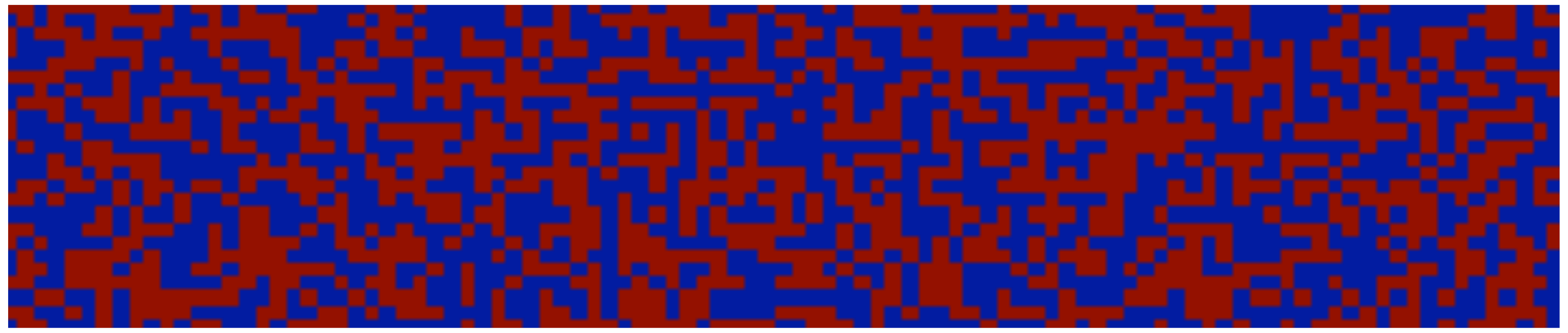
Taking Incoherent Projections of a Time Series - Bernoulli

x



Signal to be acquired

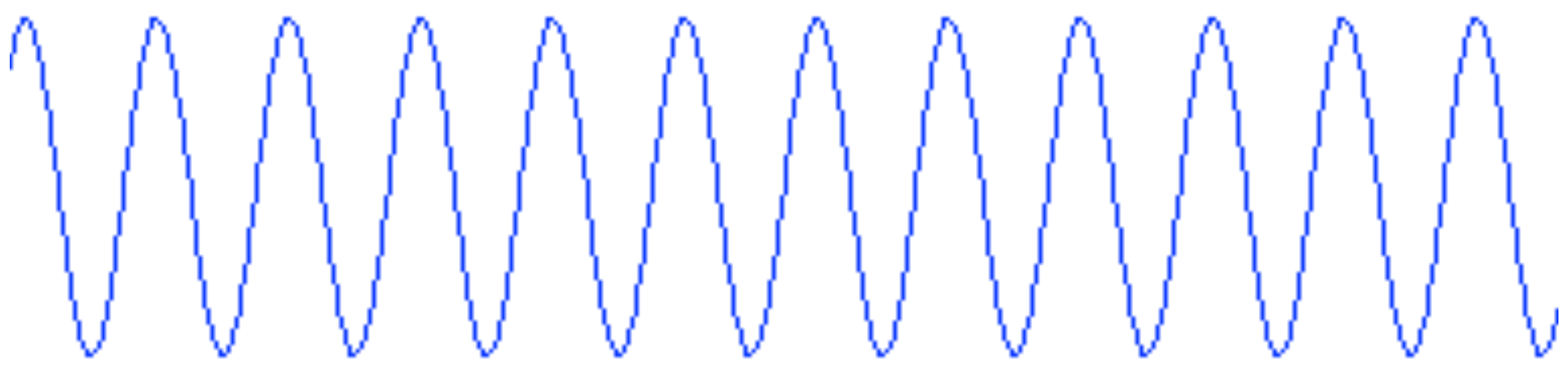
C



25 x 100 Bernoulli distributed ($p = 0.5$) random numbers $\in \{-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}\}$

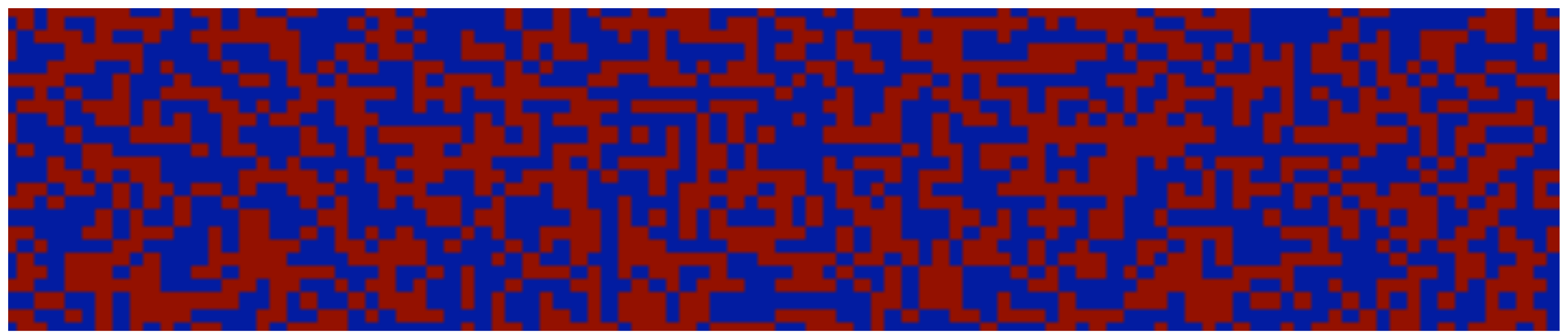
Taking Incoherent Projections of a Time Series - Bernoulli

x



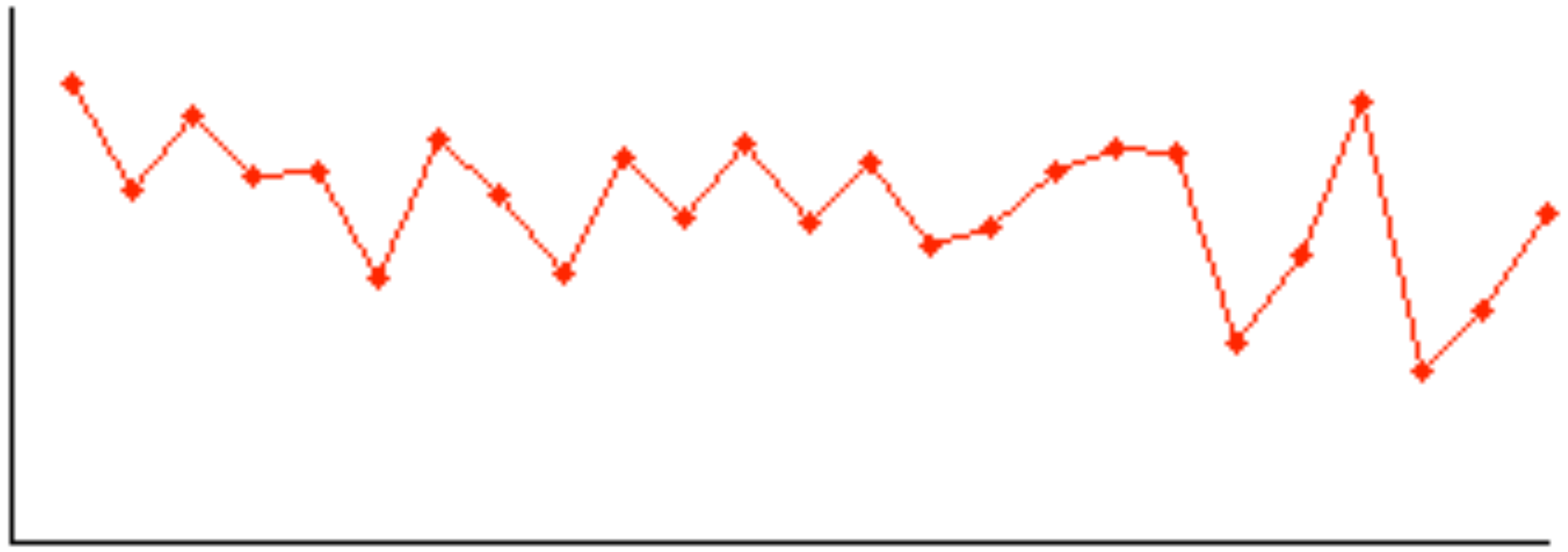
Signal to be acquired

C



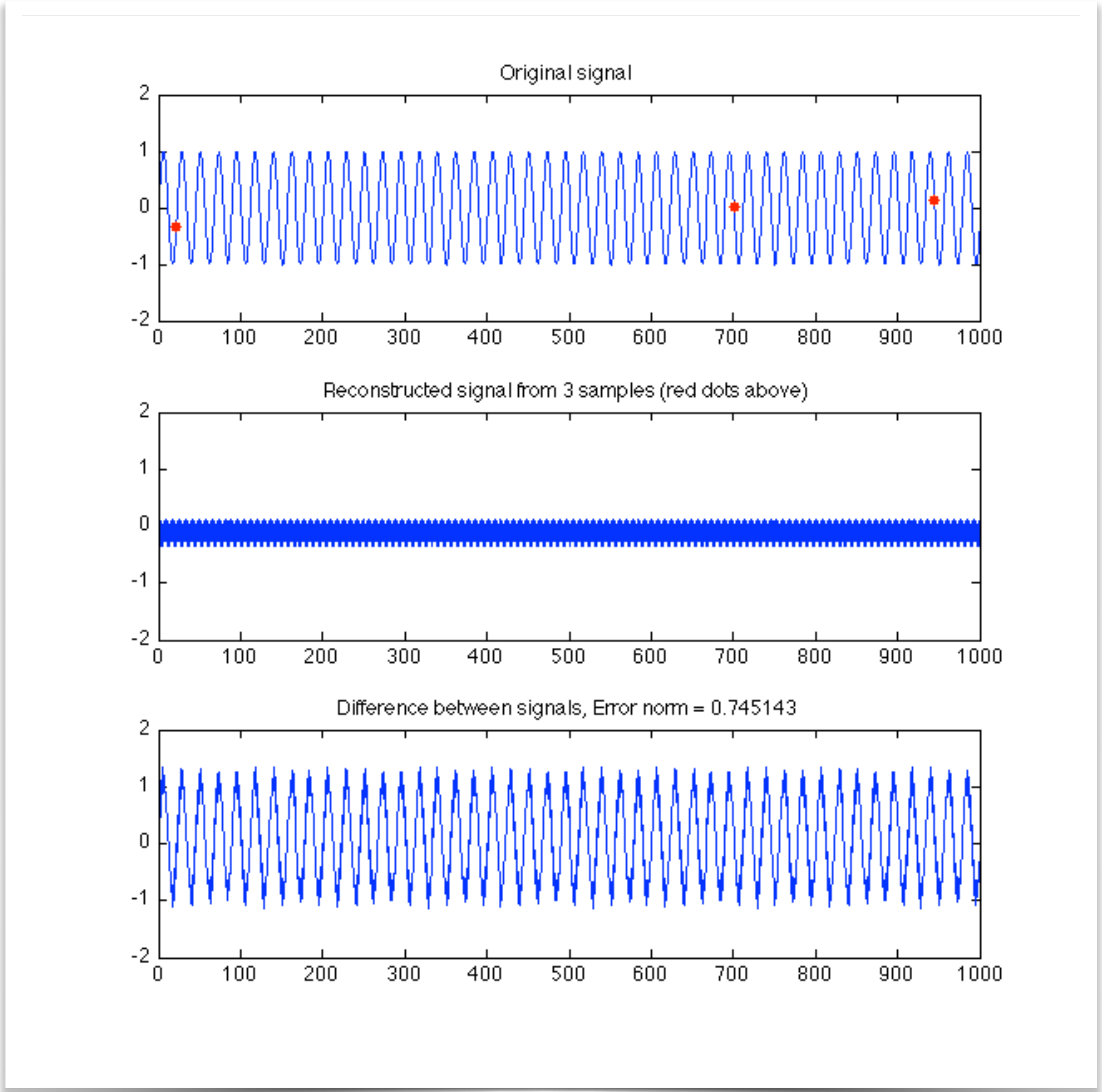
25 x 100 Bernoulli distributed ($p = 0.5$) random numbers $\in \{-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}\}$

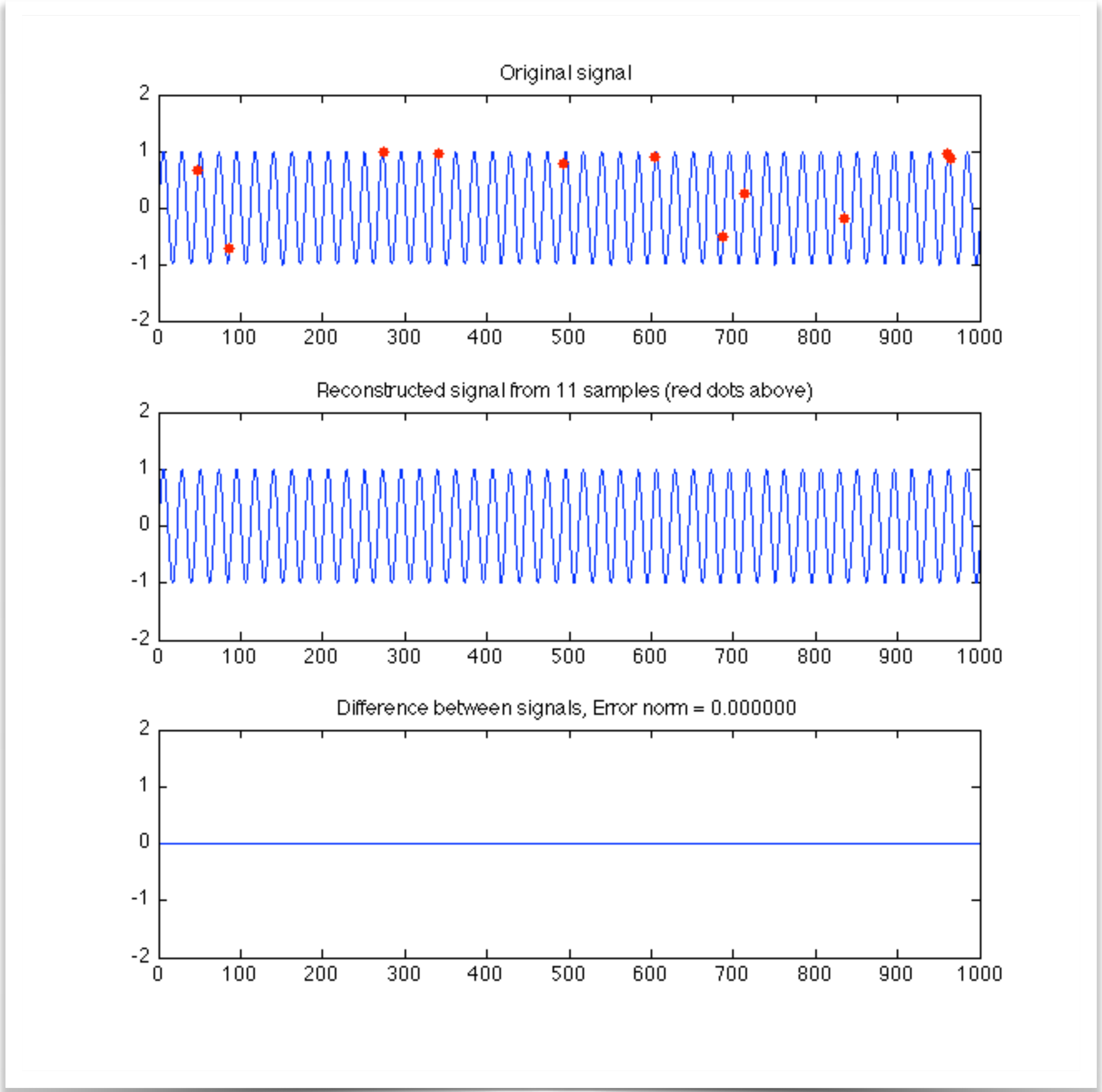
y



Measurements actually acquired

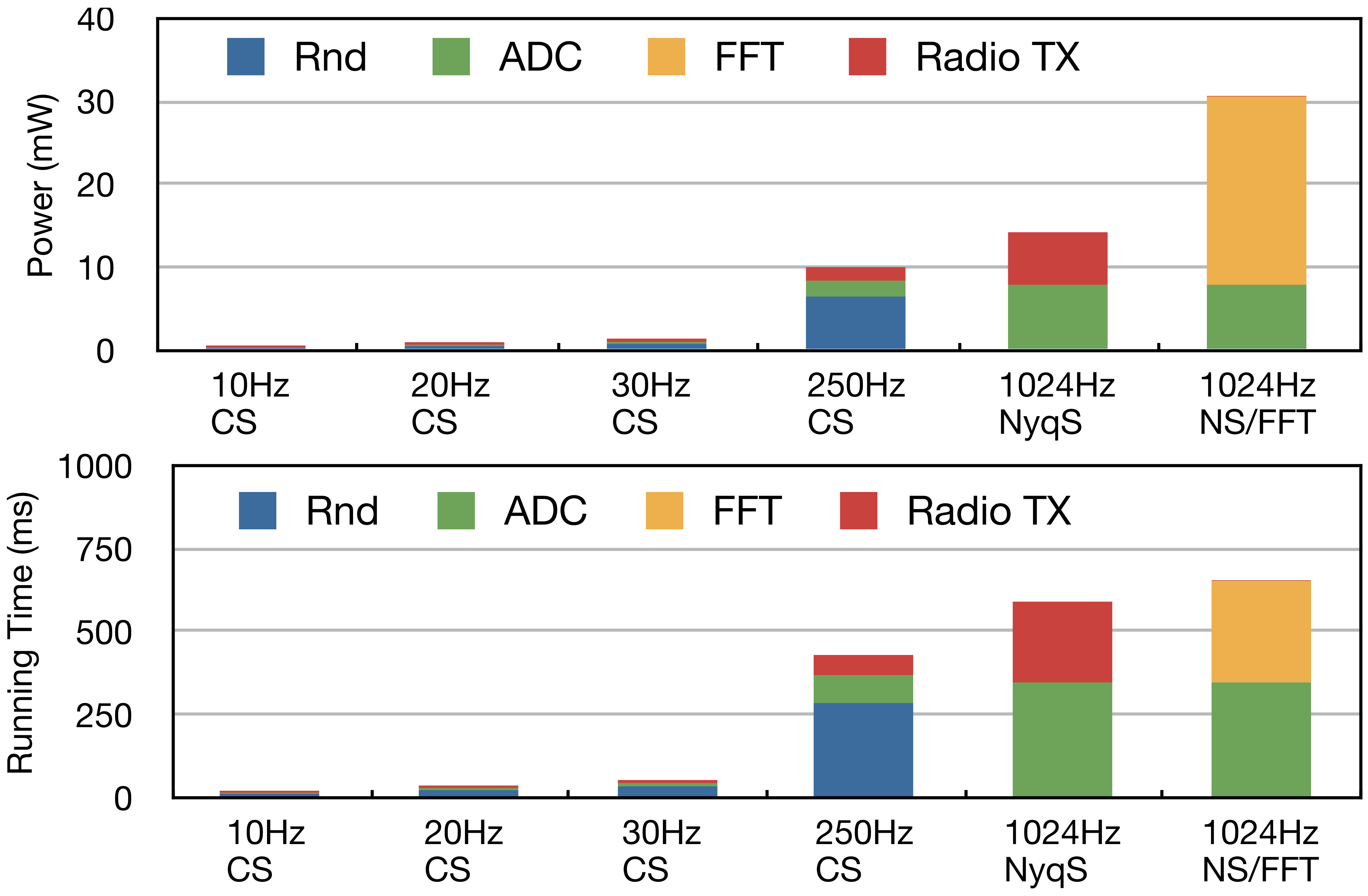
Compressed Sensing Example





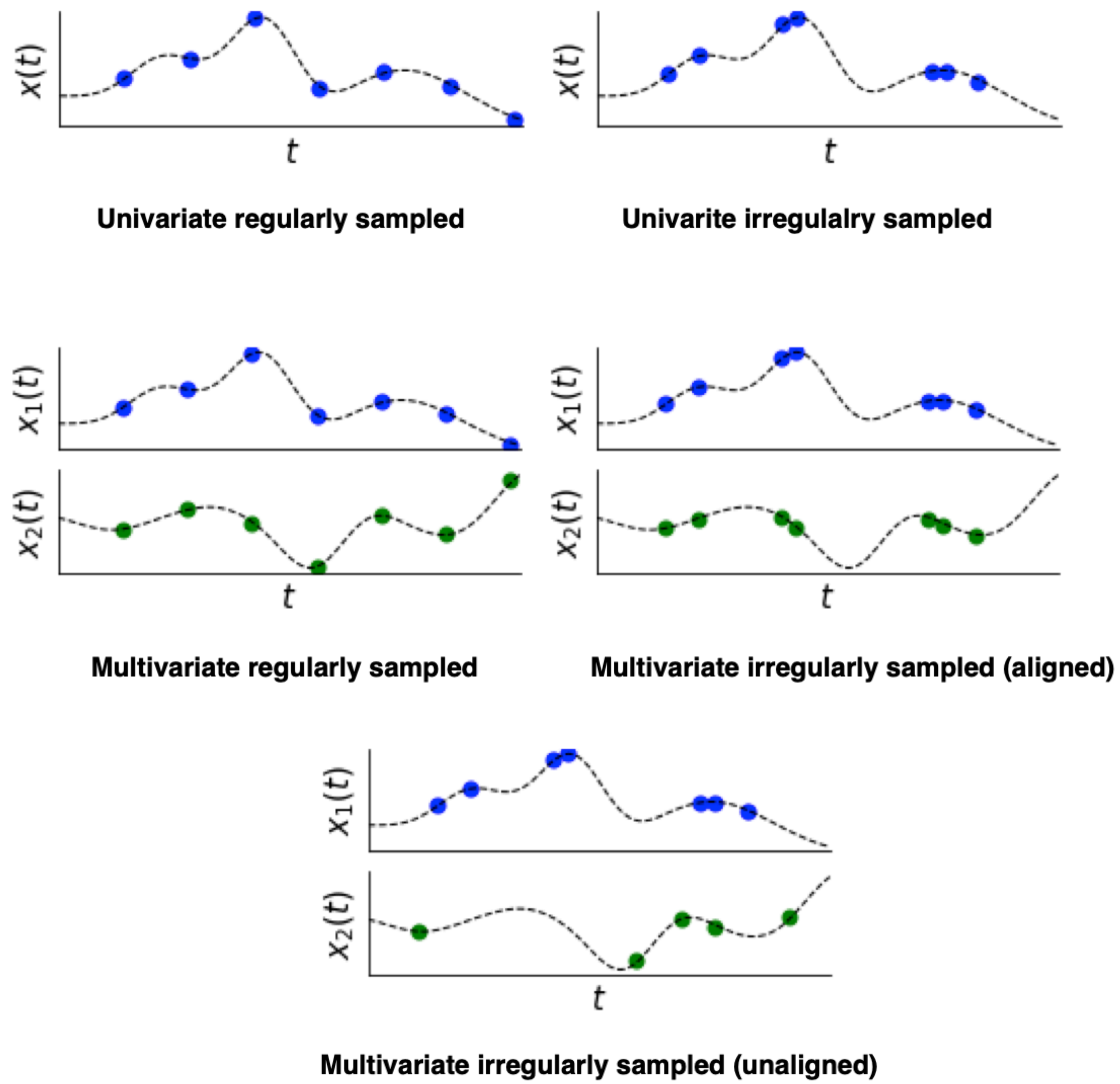
Compressive Sampling to Save Energy on Edge Devices

- ADC and associated analog amplifier may be power hungry
- Lossy compression may be power hungry (doing FFT, DCT etc.)
- In compressive sampling, the highest cost is random number generation
- To maximize benefit, may also need to duty cycle analog circuitry



Learning from Irregularly Sampled Time Series

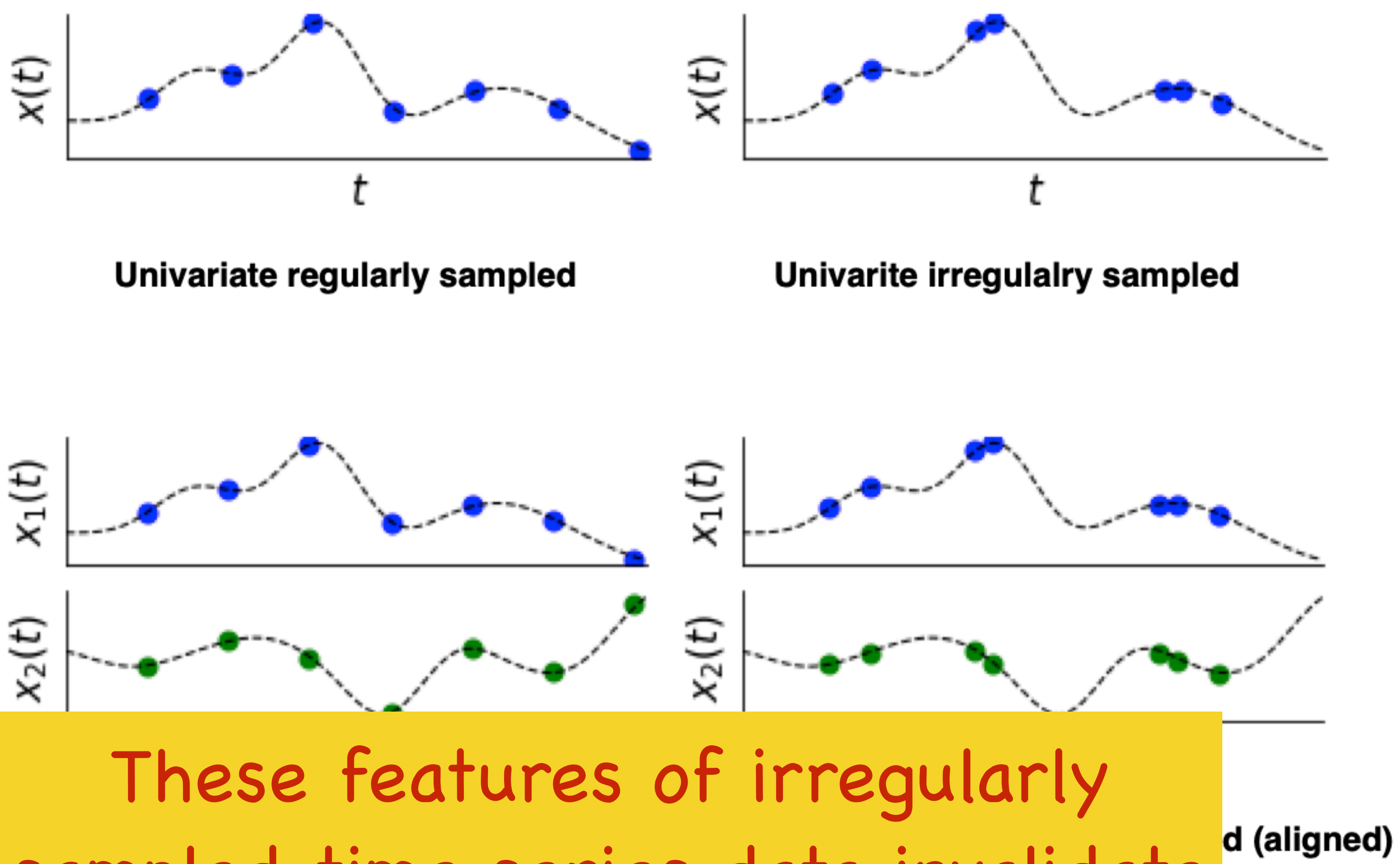
Irregularly Sampled Time Series Data



- Present fundamental challenges to many classical models from machine learning and statistics
- Consider supervised learning task where a model takes as input an irregularly sampled time series and must predict a scalar output
  - Training set $\mathscr{D}$ with samples $(\mathbf{s}_i, y_i)$ where $\mathbf{s}_i$'s are irregularly samples time series, and y_i's are corresponding labels
- Problems:
  - **Variable gaps** between successive observation time points
  - **Variable number of observations**: the total # of observations across dimensions can vary across samples
  - **Lack of alignment:** Different dimensions of a single multivariate time series can be observed at a different collection of time points. The collection of observation times across dimensions can also differ between samples.

Shukla & Marlin, https://arxiv.org/pdf/2012.00168.pdf

Irregularly Sampled Time Series Data



These features of irregularly sampled time series data invalidate the assumption of a coherent fixed-dimensional feature space, which underlies most basic supervised and un-supervised learning models.

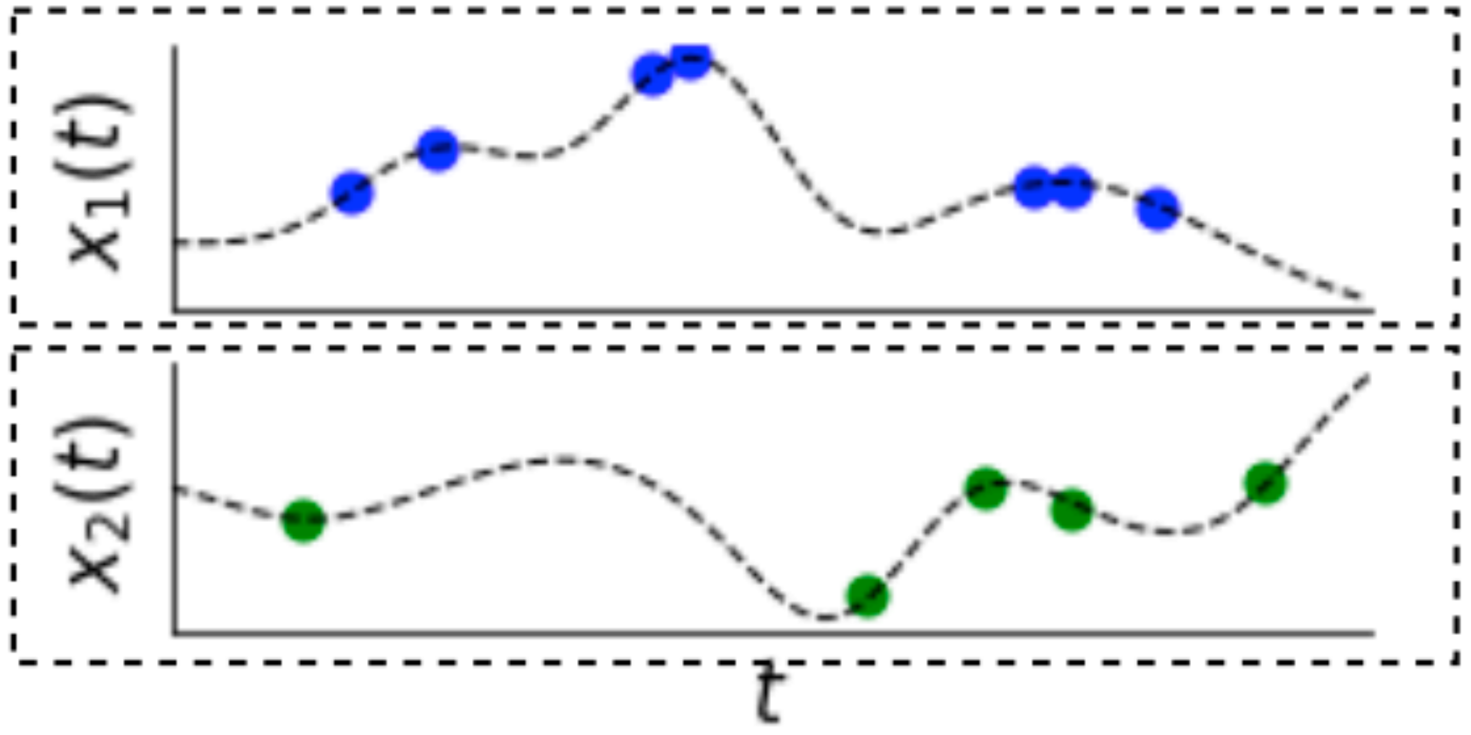
- Present fundamental challenges to many classical models from machine learning and statistics
- Consider supervised learning task where a model takes as input an irregularly sampled time series and must predict a scalar output
  - Training set $\mathscr{D}$ with samples $(\mathbf{s}_i, y_i)$ where $\mathbf{s}_i$'s are irregularly samples time series, and y_i's are corresponding labels
- Problems:
  - **Variable gaps** between successive observation time points
  - **Variable number of observations**: the total # of observations across dimensions can vary across samples
  - **Lack of alignment:** Different dimensions of a single multivariate time series can be observed at a different collection of time points. The collection of observation times across dimensions can also differ between samples.

Shukla & Marlin, *https://arxiv.org/pdf/2012.00168.pdf*

Data Representations for Irregularly Sampled Time Series

- There are several possible data representations for multivariate irregularly sampled time series.
- These representations are equivalent, but expose different properties and suggest different approaches to modeling

Series-based Representation



Multivariate irregularly sampled (unaligned)

$$\mathbf{s} = [\mathbf{s}_1, \mathbf{s}_2]$$
$$\mathbf{s}_1 = (\mathbf{t}_1, \mathbf{x}_1)$$
$$\mathbf{t}_1 = [t_{11}, \ldots, t_{L_1 1}]$$
$$\mathbf{x}_1 = [x_{11}, \ldots, x_{L_1 1}]$$
$$\mathbf{s}_2 = (\mathbf{t}_2, \mathbf{x}_2)$$
$$\mathbf{t}_2 = [t_{12}, \ldots, t_{L_2 2}]$$
$$\mathbf{x}_2 = [x_{12}, \ldots, x_{L_2 2}]$$

A D-dimensional multivariate irregularly sampled time series $\mathbf{s} = [\mathbf{s}_1, \ldots, \mathbf{s}_D]$ is represented as a collection of univariate irregularly sampled time series, one per dimension.

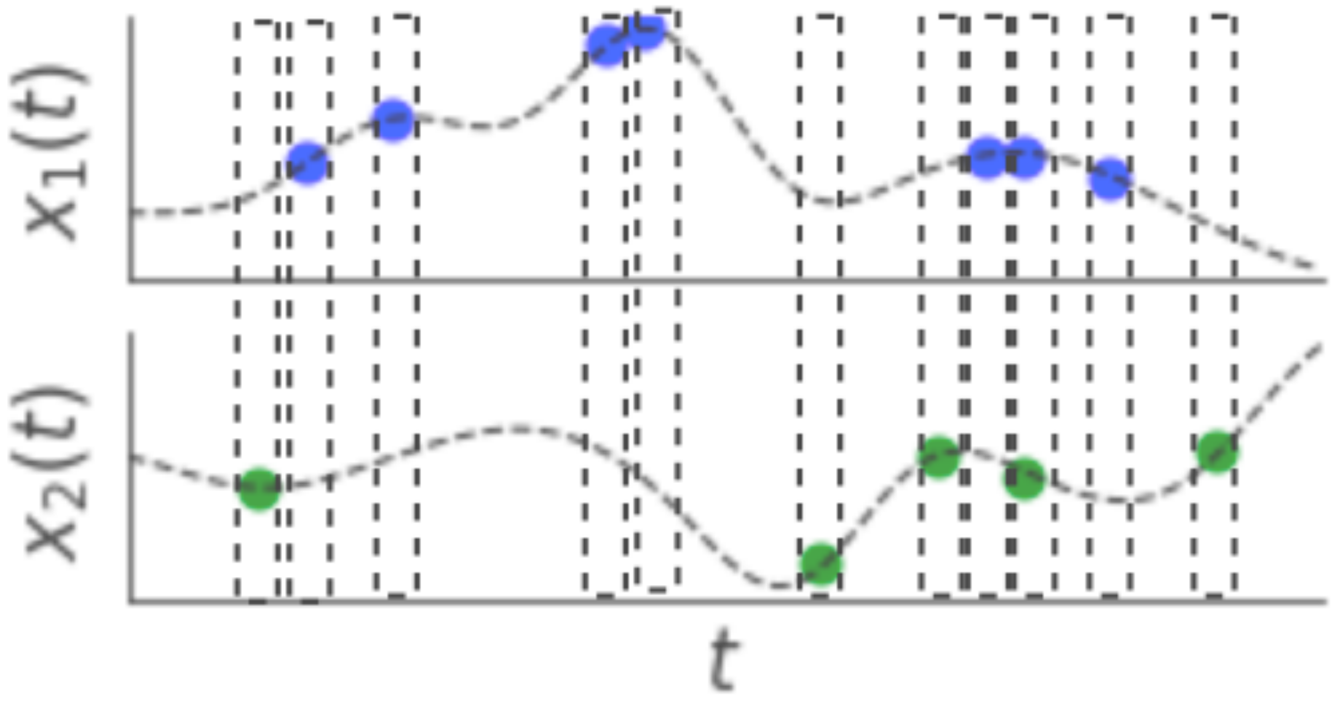
- $\mathbf{s}_d = (\mathbf{t}_d, \mathbf{x}_d)$ indicates the time series for dimension d.
- $\mathbf{t}_d$ indicates the collection of time points with observed values for dimension d.
- $\mathbf{x}_d$ indicates the corresponding collection of observed values.

Shukla & Marlin, https://arxiv.org/pdf/2012.00168.pdf

Data Representations for Irregularly Sampled Time Series

- There are several possible data representations for multivariate irregularly sampled time series.
- These representations are equivalent, but expose different properties and suggest different approaches to modeling

Vector-based Representation



Multivariate irregularly sampled (unaligned)

$$\mathbf{s} = (\mathbf{t}, \mathbf{x}, \mathbf{r})$$
$$\mathbf{t} = [t_1, \ldots, t_L]$$
$$\mathbf{x} = [\mathbf{x}_1, \cdots, \mathbf{x}_L]$$
$$\mathbf{r} = [\mathbf{r}_1, \cdots, \mathbf{r}_L]$$

$$\mathbf{x}_i = \begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix} \quad x_{id} = \begin{cases} x_{id}, & \text{if } x_{id} \text{ is observed} \\ \text{NA}, & \text{otherwise} \end{cases}$$

$$\mathbf{r}_i = \begin{bmatrix} r_{i1} \\ r_{i2} \end{bmatrix} \quad r_{id} = \begin{cases} 1, & \text{if } x_{id} \text{ is observed} \\ 0, & \text{otherwise} \end{cases}$$

In this representation, there is a single collection of time points $\mathbf{t}$. At each time point t_i, there is a D-dimensional vector-valued observation $\mathbf{x}_i$.

In the general case, not all dimensions of $\mathbf{x}_i$ are observed, leading to the need to explicitly represent which dimensions are observed and which are missing.

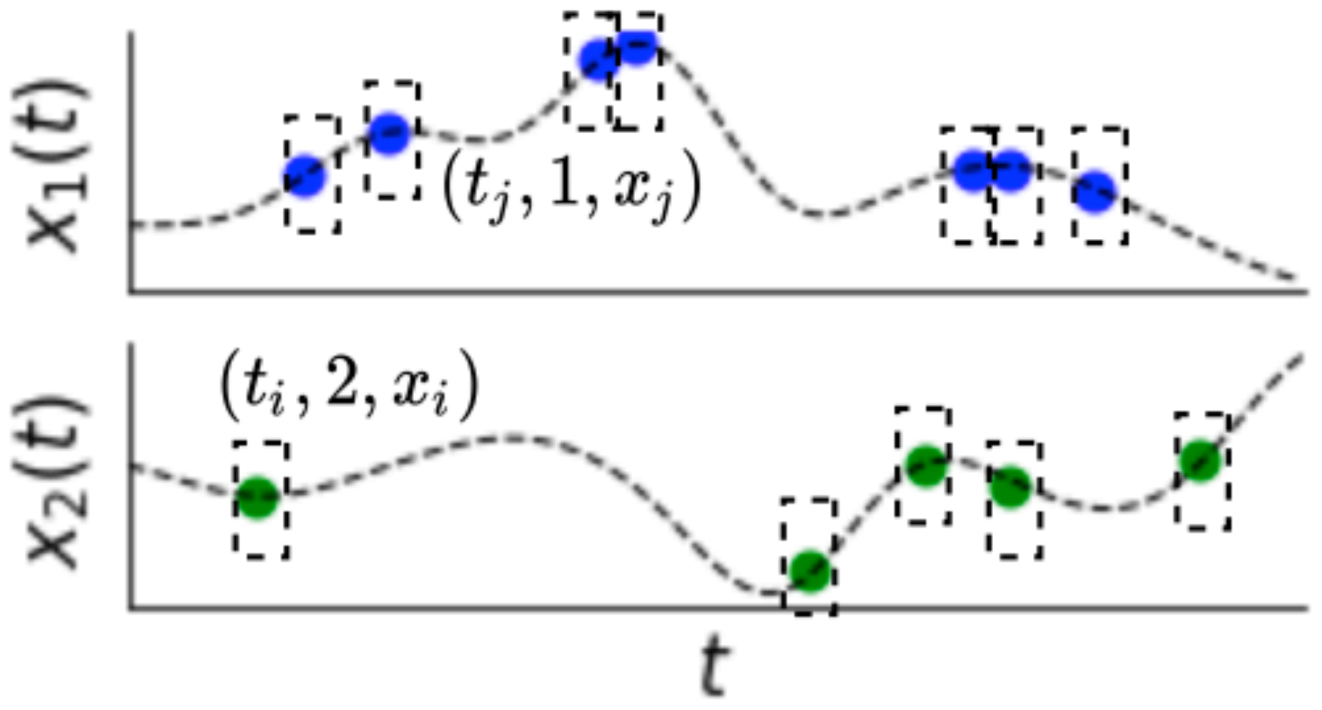
A D-dimensional binary response indicator vector $\mathbf{r}_i$ at each time point t_i indicates which dimensions are observed and which are missing.

Shukla & Marlin, https://arxiv.org/pdf/2012.00168.pdf

Data Representations for Irregularly Sampled Time Series

- There are several possible data representations for multivariate irregularly sampled time series.
- These representations are equivalent, but expose different properties and suggest different approaches to modeling

Set-based Representation



Multivariate irregularly sampled (unaligned)

$$\mathbf{s} = \{(t_1, 2, x_1), (t_2, 1, x_2), (t_3, 1, x_3), \cdots\}$$

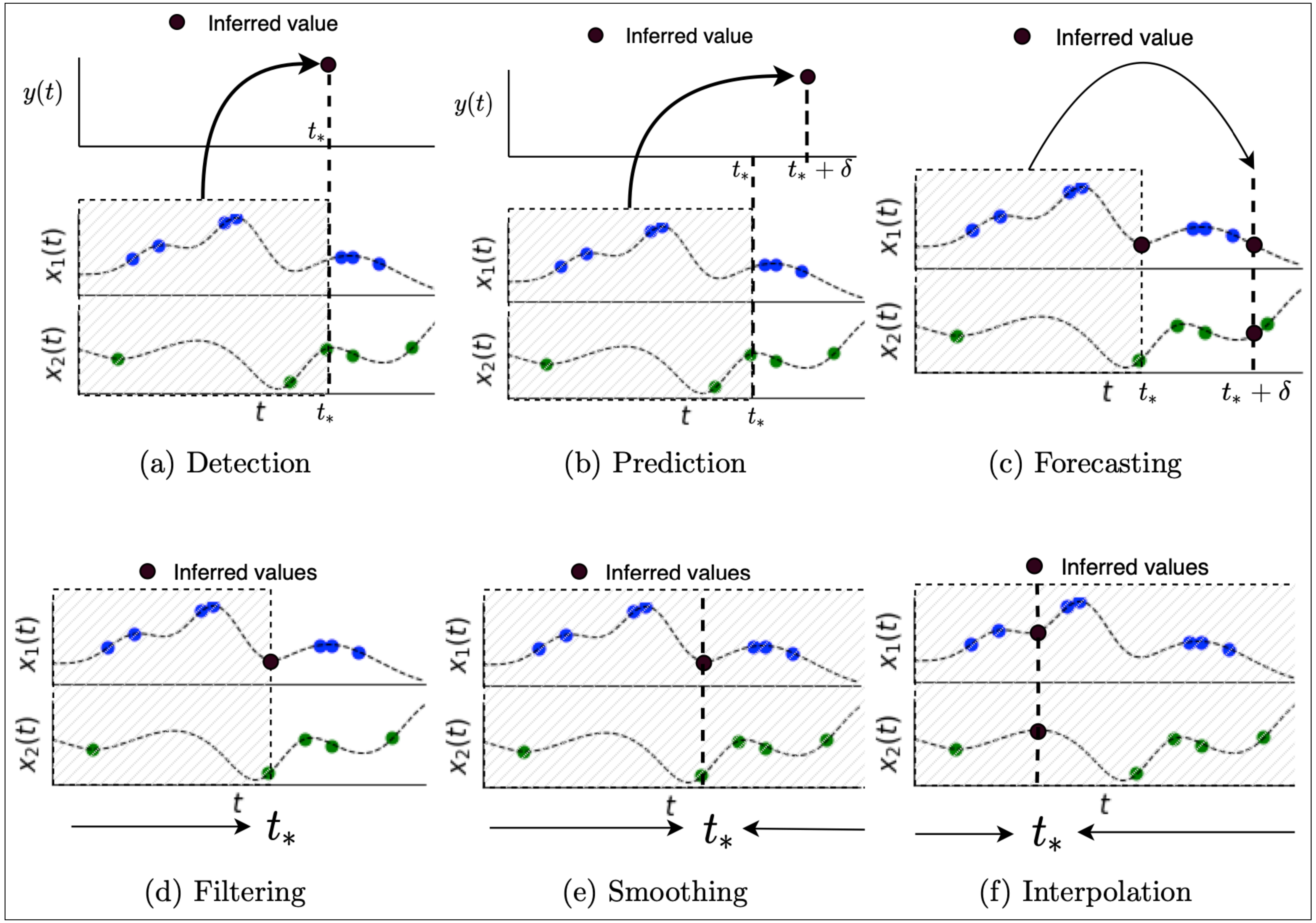
In this representation, a D-dimensional multivariate irregularly sampled time series is represented as a set of (*time*, *dimension*, *value*) tuples, one for each observation.

Shukla & Marlin, https://arxiv.org/pdf/2012.00168.pdf

Inference Tasks

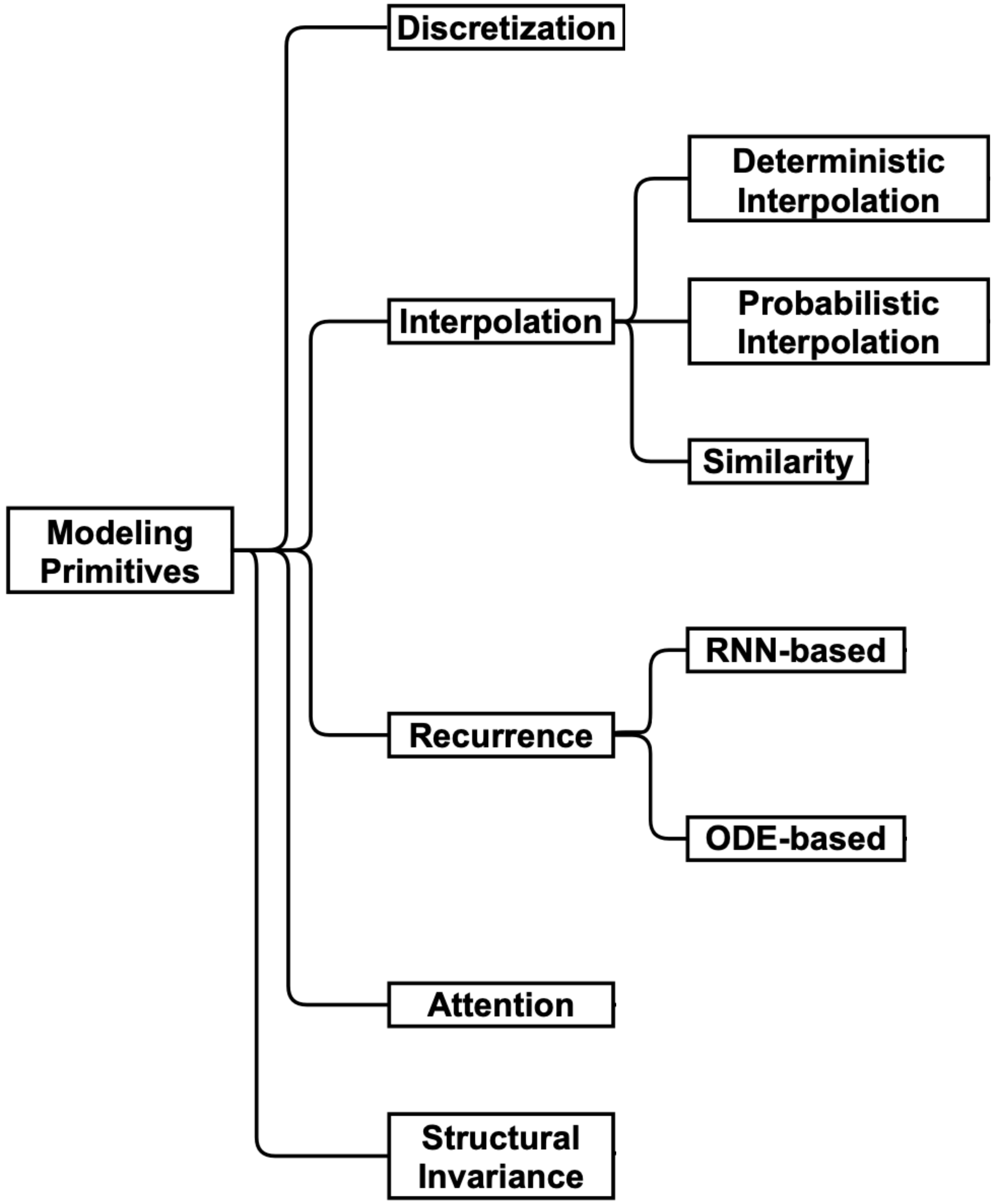
Note that in machine learning, the inference for any quantity that is not known is often referred to as a "prediction", but here we use the term "prediction" to refer to a task where the inference is for the value of the output variable at a time that is in the relative future of the time point t_* at which the inference is made.

- **Detection:** Inferring prediction target values $\mathbf{y}[t_*]$ at time t_* conditioning on the observations $\mathbf{s}[:t_*]$ available up to and including time t_*.
- **Prediction:** Inferring prediction target values $\mathbf{y}[t_* + \delta]$ at time $t_* + \delta$ (for $\delta > 0$) conditioning on the observations $\mathbf{s}[:t_*]$ available up to and including time t_*
- **Forecasting:** Inferring $\mathbf{x}[t_* + \delta]$ (for $\delta > 0$) by conditioning on the observations $\mathbf{s}[:t_*]$ up to an including time t_*.
- **Filtering:** Inferring missing variables $\mathbf{x}^m[t_*]$ at time t_* by conditioning on the observations $\mathbf{s}[:t_*]$ up to an including time t_*.
- **Smoothing:** Inferring the values of $\mathbf{x}^m[t_*]$ at time t_* using the observed data in $\mathbf{s}$.
- **Interpolation:** Inferring the values of $\mathbf{x}[t_*]$ at time t_* using the observed data in $\mathbf{s}$.



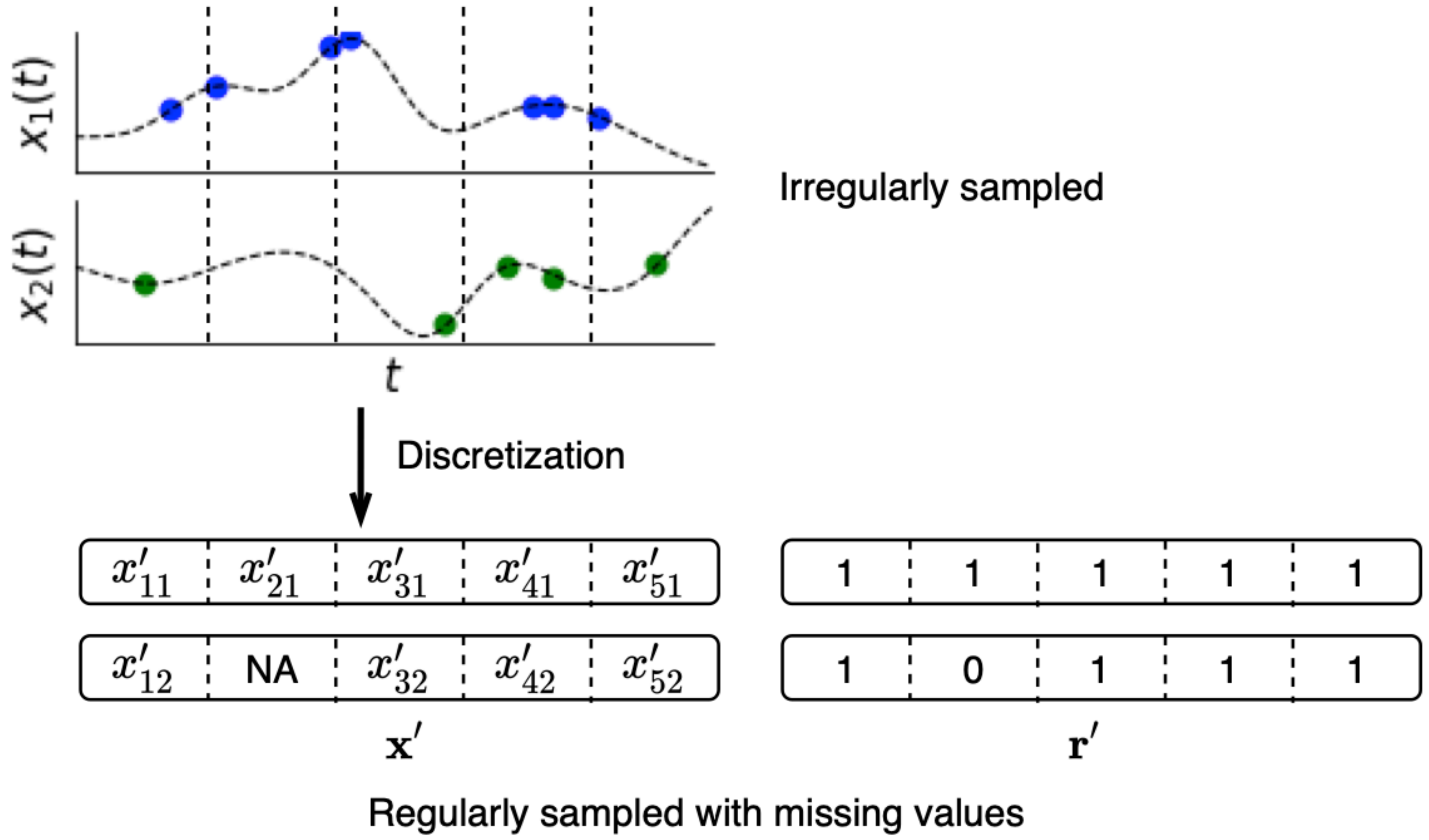
Shukla & Marlin, https://arxiv.org/pdf/2012.00168.pdf

Modeling Primitives for Irregularly Sampled Time Series



Shukla & Marlin, https://arxiv.org/pdf/2012.00168.pdf

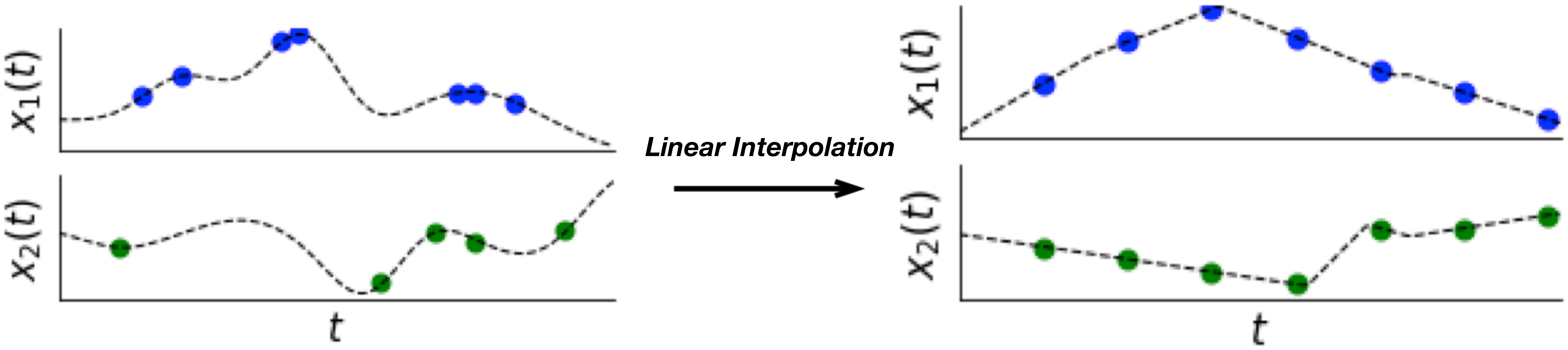
Discretization



- Reduces to a regularly sampled multivariate time series with missing values
  - vector-based representation with missing data indicator
- Approach: divide the time axis into equal sized non-overlapping intervals and define a value within each time interval based on the observed values falling within that interval
  - e.g. average or median

Shukla & Marlin, https://arxiv.org/pdf/2012.00168.pdf

Interpolation

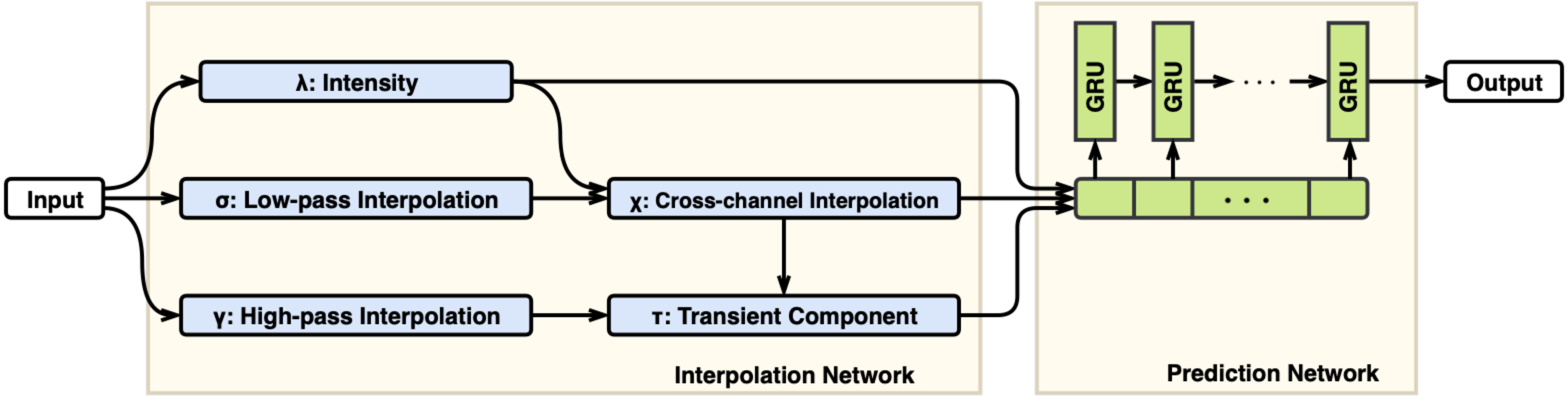


- Approach:
  1. Define a set of K reference time points $\tau = [\tau_1, \ldots, \tau_K]$
  2. Use a a basic kernel smoother to produce interpolated values at the reference time points
     - kernel typically puts higher weights (learnable parameters) to points that are closer to the reference points
     - for multivariate case, can account for both correlation in time and correlation across different dimensions
     - deterministic: e.g., squared exponential kernel, stochastic: e.g., Gaussian process regression

Shukla & Marlin, https://arxiv.org/pdf/2012.00168.pdf

Example: Interpolation-Prediction Networks (IP-Net)

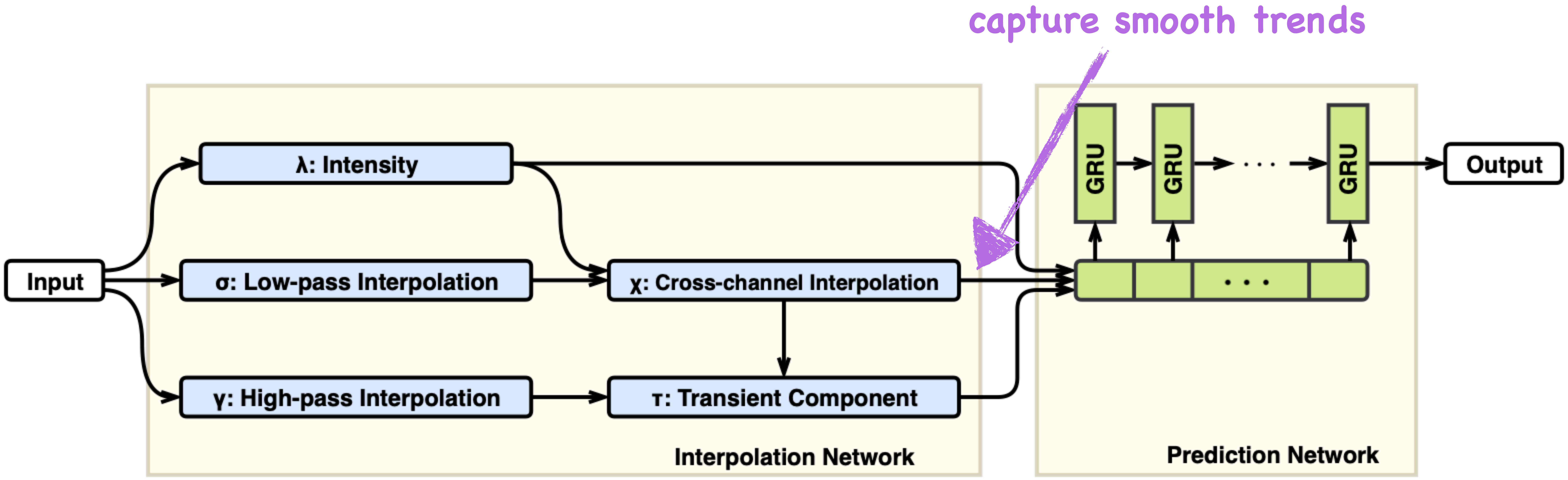
- Applies multiple semi-parametric interpolation schemes to obtain a regularly-sampled time series representation with samples at a set of reference time points
  - The parameters of the interpolation network are trained with the classifier in an end-to-end setup
- Prediction Network can be any standard supervised neural network architecture such as fully-connected feedforward, convolutional, or recurrent network.



Shukla, Satya Narayan, and Benjamin M. Marlin. "Interpolation-prediction networks for irregularly sampled time series." arXiv preprint arXiv:1909.07782 (2019). https://arxiv.org/pdf/1909.07782

Example: Interpolation-Prediction Networks (IP-Net)

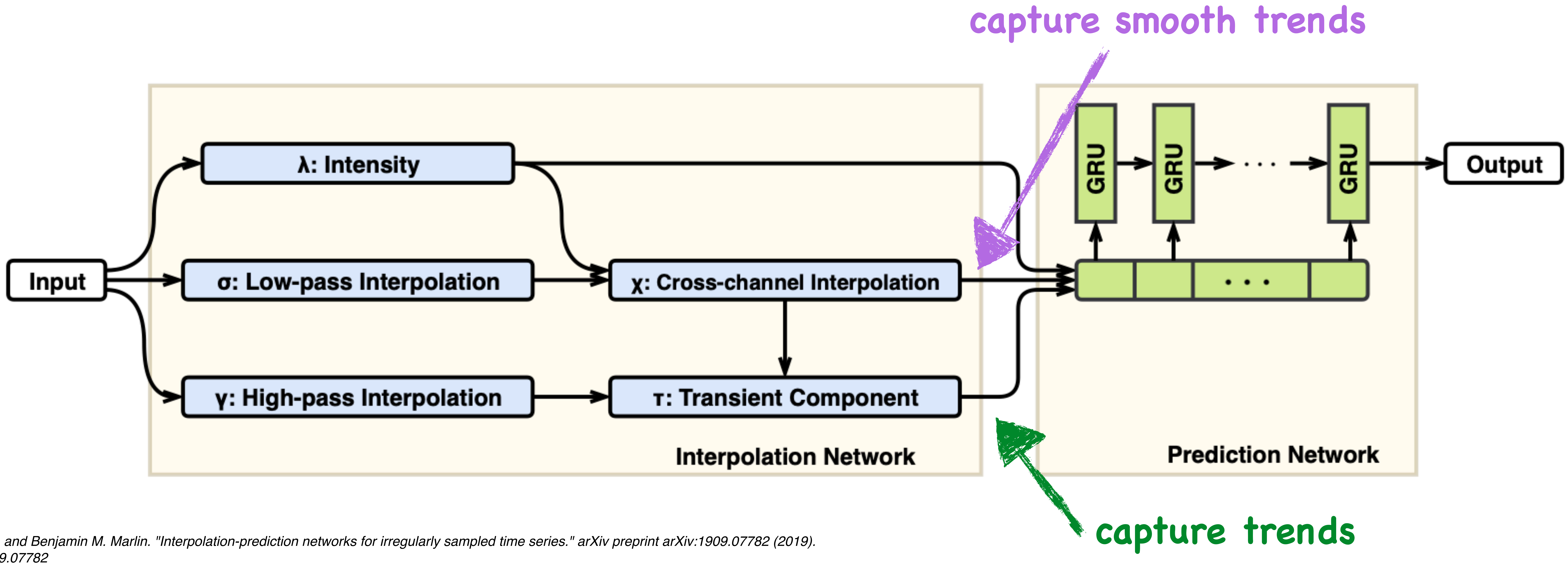
- Applies multiple semi-parametric interpolation schemes to obtain a regularly-sampled time series representation with samples at a set of reference time points
  - The parameters of the interpolation network are trained with the classifier in an end-to-end setup
- Prediction Network can be any standard supervised neural network architecture such as fully-connected feedforward, convolutional, or recurrent network.



Shukla, Satya Narayan, and Benjamin M. Marlin. "Interpolation-prediction networks for irregularly sampled time series." arXiv preprint arXiv:1909.07782 (2019). https://arxiv.org/pdf/1909.07782

Example: Interpolation-Prediction Networks (IP-Net)

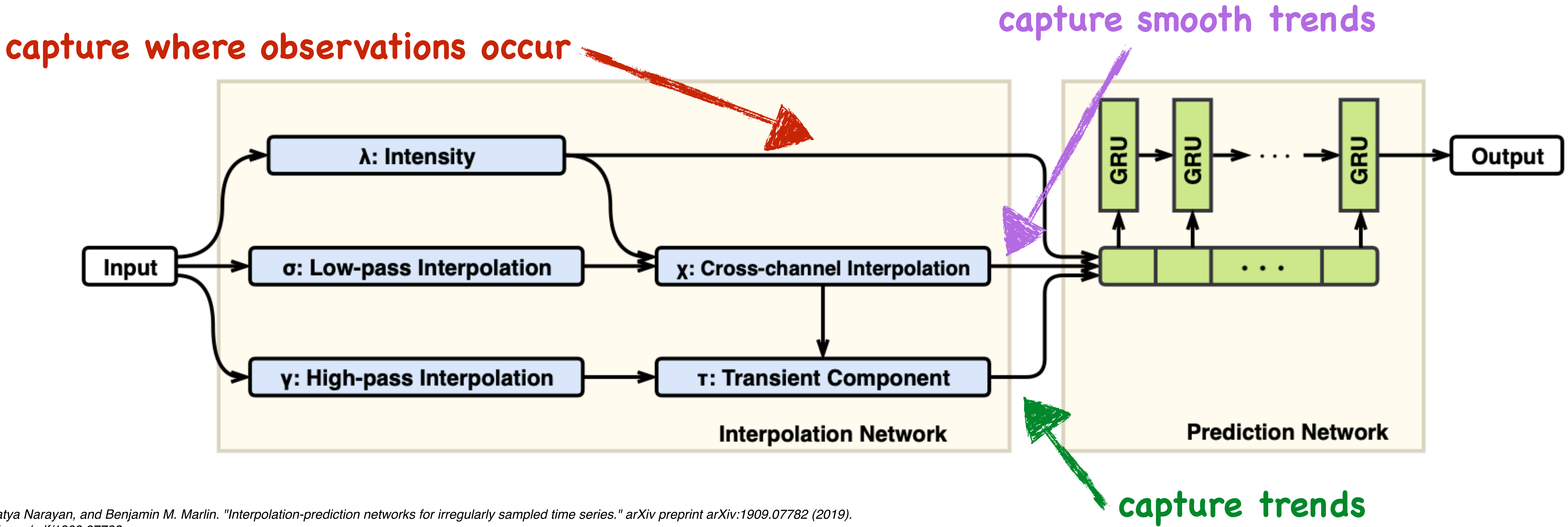
- Applies multiple semi-parametric interpolation schemes to obtain a regularly-sampled time series representation with samples at a set of reference time points
  - The parameters of the interpolation network are trained with the classifier in an end-to-end setup
- Prediction Network can be any standard supervised neural network architecture such as fully-connected feedforward, convolutional, or recurrent network.



Shukla, Satya Narayan, and Benjamin M. Marlin. "Interpolation-prediction networks for irregularly sampled time series." arXiv preprint arXiv:1909.07782 (2019). https://arxiv.org/pdf/1909.07782

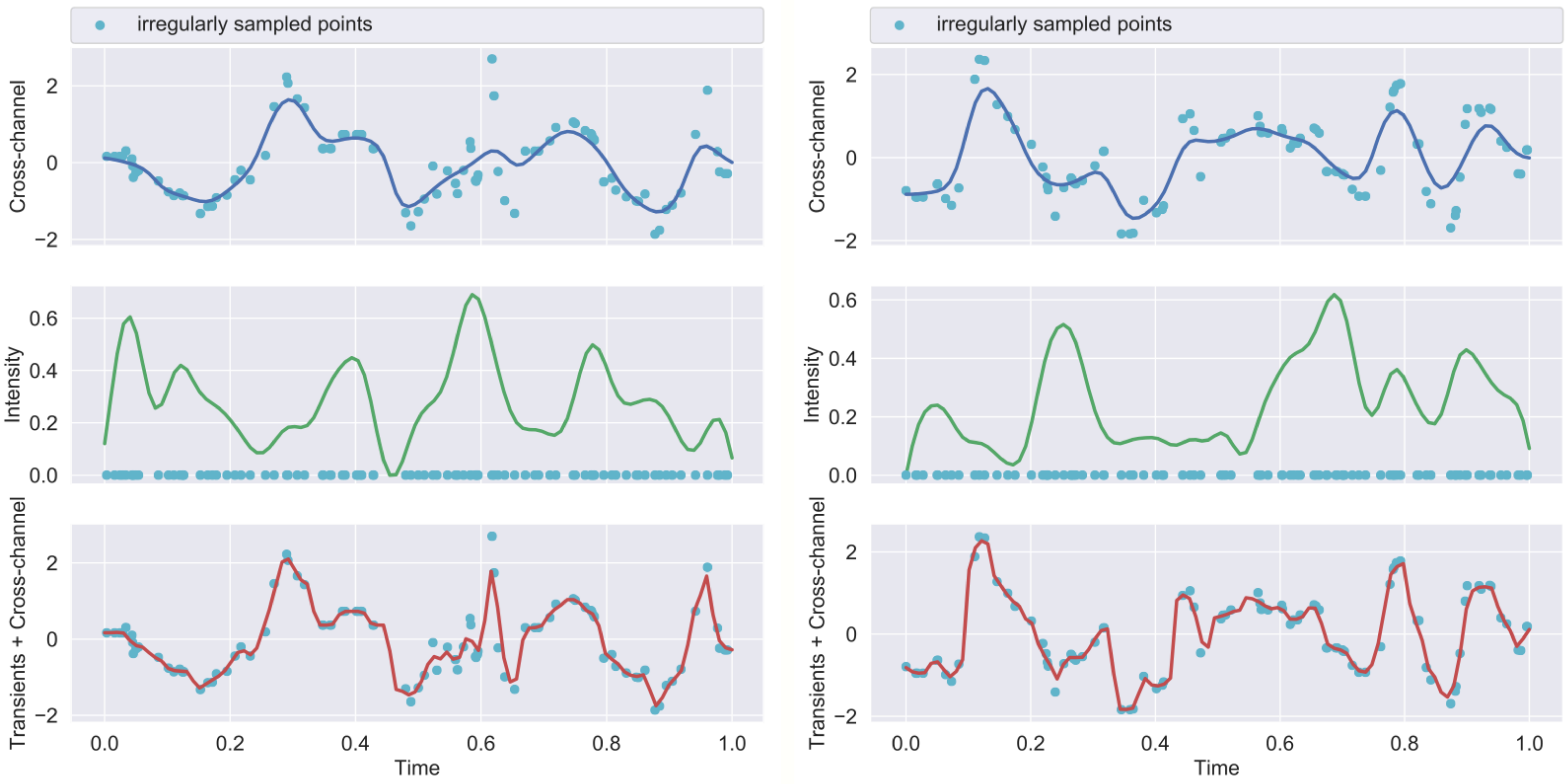
Example: Interpolation-Prediction Networks (IP-Net)

- Applies multiple semi-parametric interpolation schemes to obtain a regularly-sampled time series representation with samples at a set of reference time points
  - The parameters of the interpolation network are trained with the classifier in an end-to-end setup
- Prediction Network can be any standard supervised neural network architecture such as fully-connected feedforward, convolutional, or recurrent network.



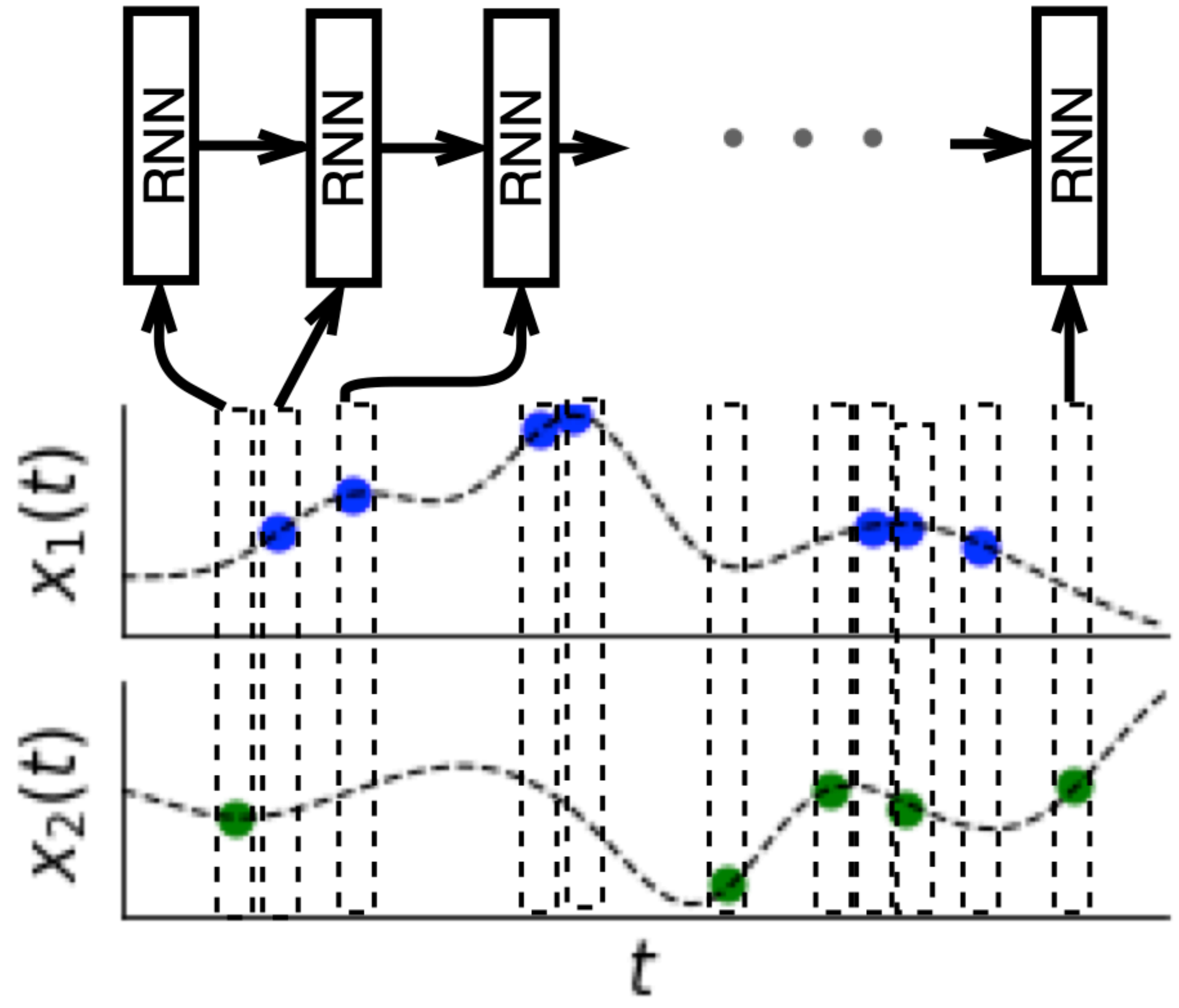
Shukla, Satya Narayan, and Benjamin M. Marlin. "Interpolation-prediction networks for irregularly sampled time series." arXiv preprint arXiv:1909.07782 (2019). https://arxiv.org/pdf/1909.07782

Output of Interpolation Network



Shukla, Satya Narayan, and Benjamin M. Marlin. "Interpolation-prediction networks for irregularly sampled time series." arXiv preprint arXiv:1909.07782 (2019). https://arxiv.org/pdf/1909.07782

Recurrence



Multivariate irregularly sampled (unaligned)

- Use a RNN cell with the ability to explicitly represent time to integrate the input at each time point with the latent state from the previous time point
  - e.g. append the time points or inter-observation intervals to the vector-valued observations $[\mathbf{x}_{in}, t_{in}]$ or $[\mathbf{x}_{in}, t_{in} - t_{i-1n}]$

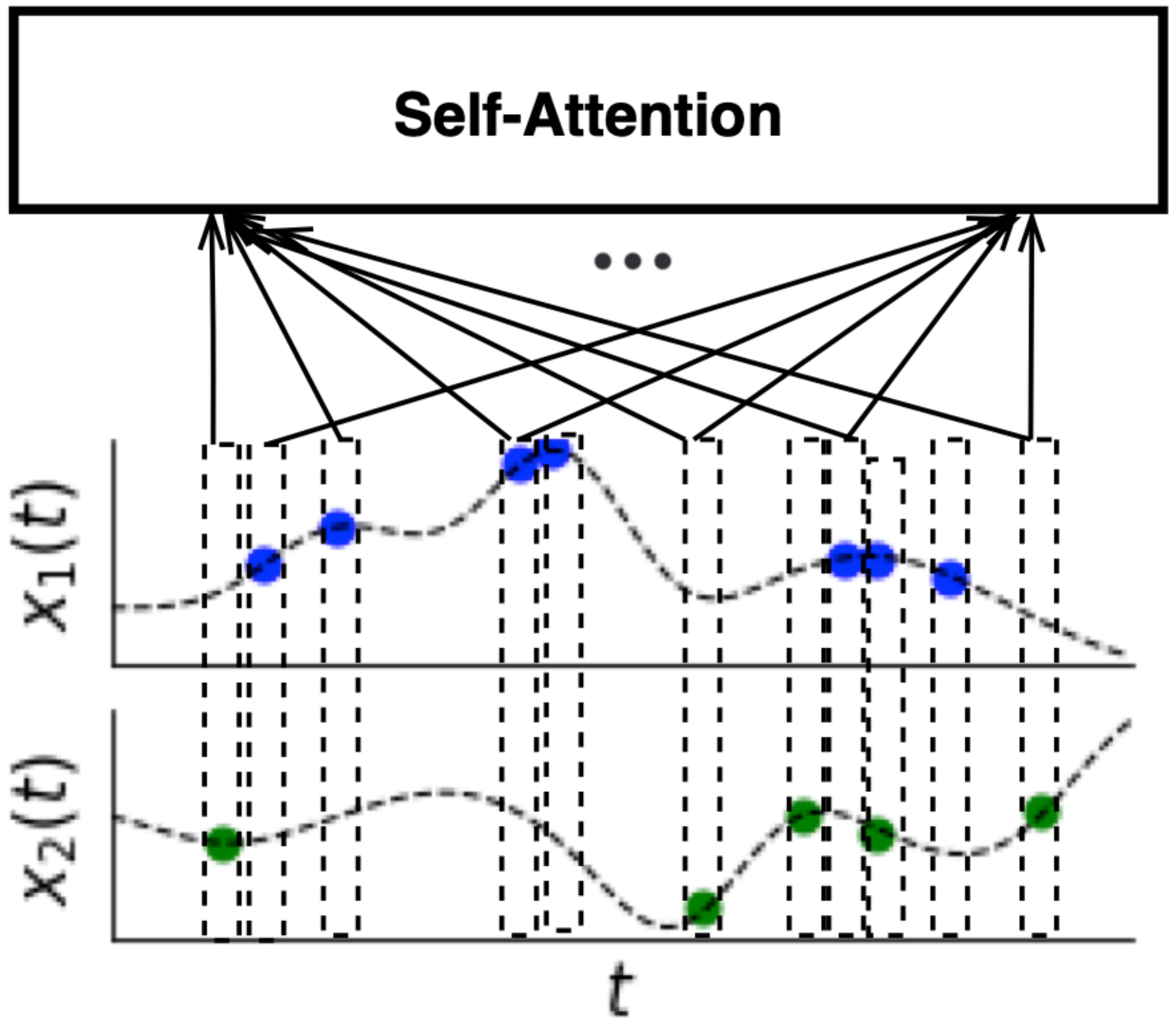
$$\mathbf{h}_i = f_\theta(\mathbf{h}_{i-1}, \mathbf{z}_{in})$$
$$\hat{\mathbf{y}}_{in} = q_\phi(\mathbf{h}_i)$$

- Recent work on ordinary differential equation (ODE) models in ML provides an alternative recurrence-based solution
  - In these ODE-RNN models, ODEs are used to evolve the hidden state between continuous time observations.
  - Better properties than traditional RNNs in terms of their ability to accommodate irregularly sampled data.

$$\mathbf{h}'_i = \text{ODESolve}(g_\gamma, \mathbf{h}_{i-1}, (t_{i-1n}, t_{in}))$$
$$\mathbf{h}_i = f_\theta(\mathbf{h}'_i, \mathbf{x}_{in})$$

Shukla & Marlin, https://arxiv.org/pdf/2012.00168.pdf

Attention



Multivariate irregularly sampled (unaligned)

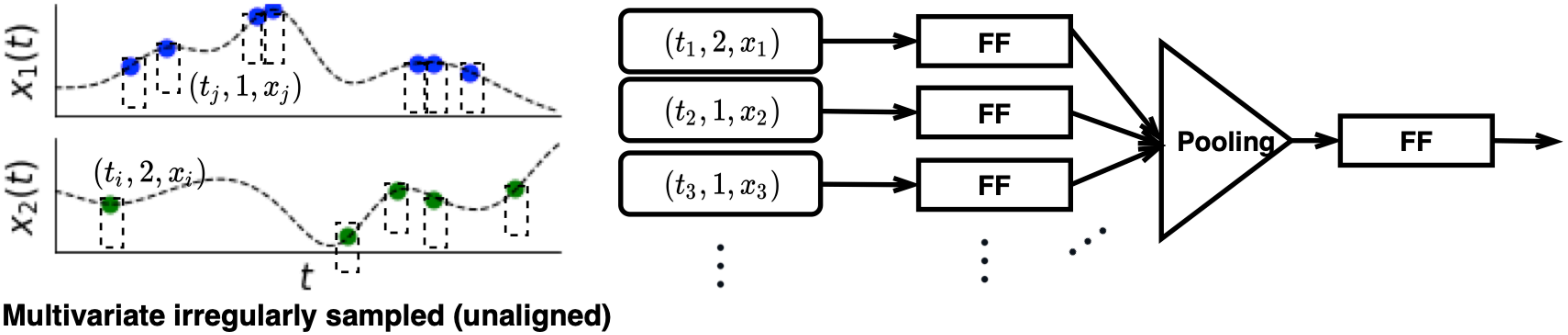
- Self-Attention module learns which regions of an input time series to attend to when computing outputs at different points in time by leveraging positional or time encodings.

$$\mathrm{Attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\,\mathbf{K}^T}{\sqrt{C}}\right)\mathbf{V}$$

- There is no recurrent structure, which allows parallel processing go the entire time series instead of sequentially as in RNN
- Time values can be converted into a vector representation using positional encoding and concatenated with the observation value (as in RNN)
- Missing values in vector-valued observations are also problematic for attention-based modules, which (like standard RNNs) expect fully observed vectors as input
  - imputation solutions can be used

Shukla & Marlin, https://arxiv.org/pdf/2012.00168.pdf

Structural Invariance



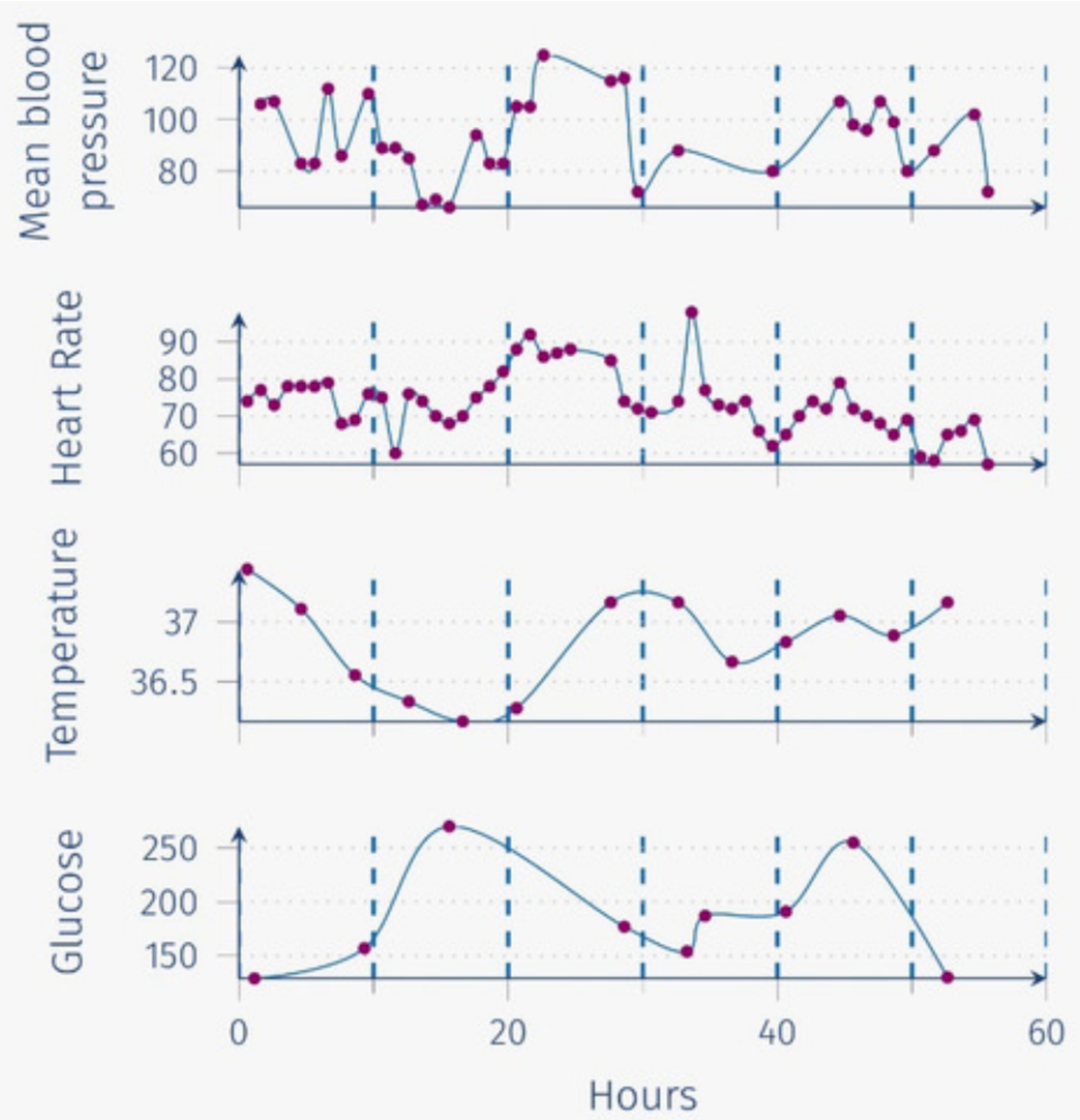
- Inspired by the set-based view of a multivariate irregularly sampled time series $\mathbf{s}_n = (t_{in}, d_{in}, \mathbf{x}_{in}) \,|\, 1 \leq i \leq L_n$
- Set-based neural network approach processes individual (time, value, dimension) tuples via an encoding function and then pools over the output of all such tuples
  - such approaches (i) produce an encoding of an input irregularly sampled time series by applying an initial encoder f_θ to individual *time-dimension-value* tuples, (ii) then perform a pooling operation (e.g. max, mean and sum) over all initial encodings in a way that is completely invariant to the temporal structure of the data, and (iii) map the output of pooling through one additional set of encoding layers g_φ to produce the final representation

$$\mathbf{h} = g_\varphi(\text{pool}(f_\theta(t_{in}, d_{in}, \mathbf{x}_{in}) \,|\, 1 \leq i \leq L_n)$$

Shukla & Marlin, https://arxiv.org/pdf/2012.00168.pdf

Example Approach: Set Functions for Time Series (SeFT)

Medical Time Series



Challenge
Imputation requires solving the harder problem of learning the time series dynamics, and also sacrifices interpretability of the inference.

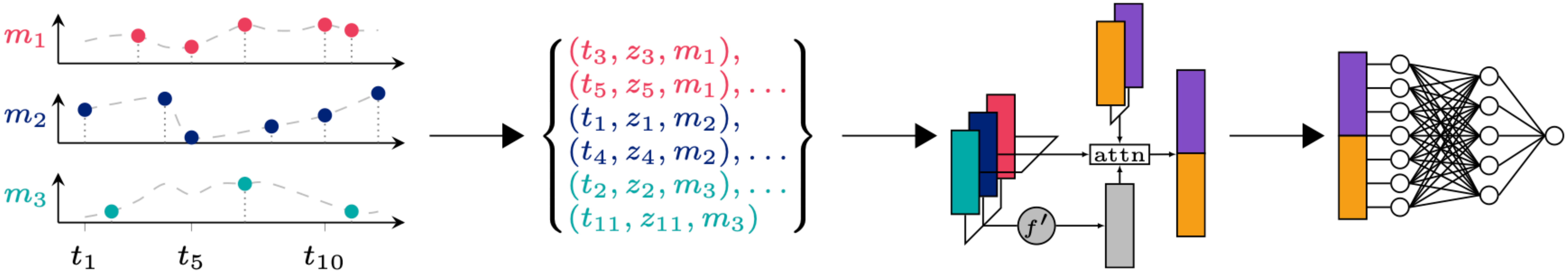
Problem Statement
Can we learn classification models on irregularlysampled time series without prior imputation?

Set Functions for Time Series
➡ Time series classification as set classification

(based on recent advances in differentiable set function learning)

Horn, Max, Michael Moor, Christian Bock, Bastian Rieck, and Karsten Borgwardt. "Set functions for time series." In International Conference on Machine Learning, pp. 4353-4363. PMLR, 2020.
http://proceedings.mlr.press/v119/horn20a/horn20a.pdf & https://slideslive.com/38928275/set-functions-for-time-series

SeFT Architecture Overview



Irregularly Sampled Multivariate Time Series

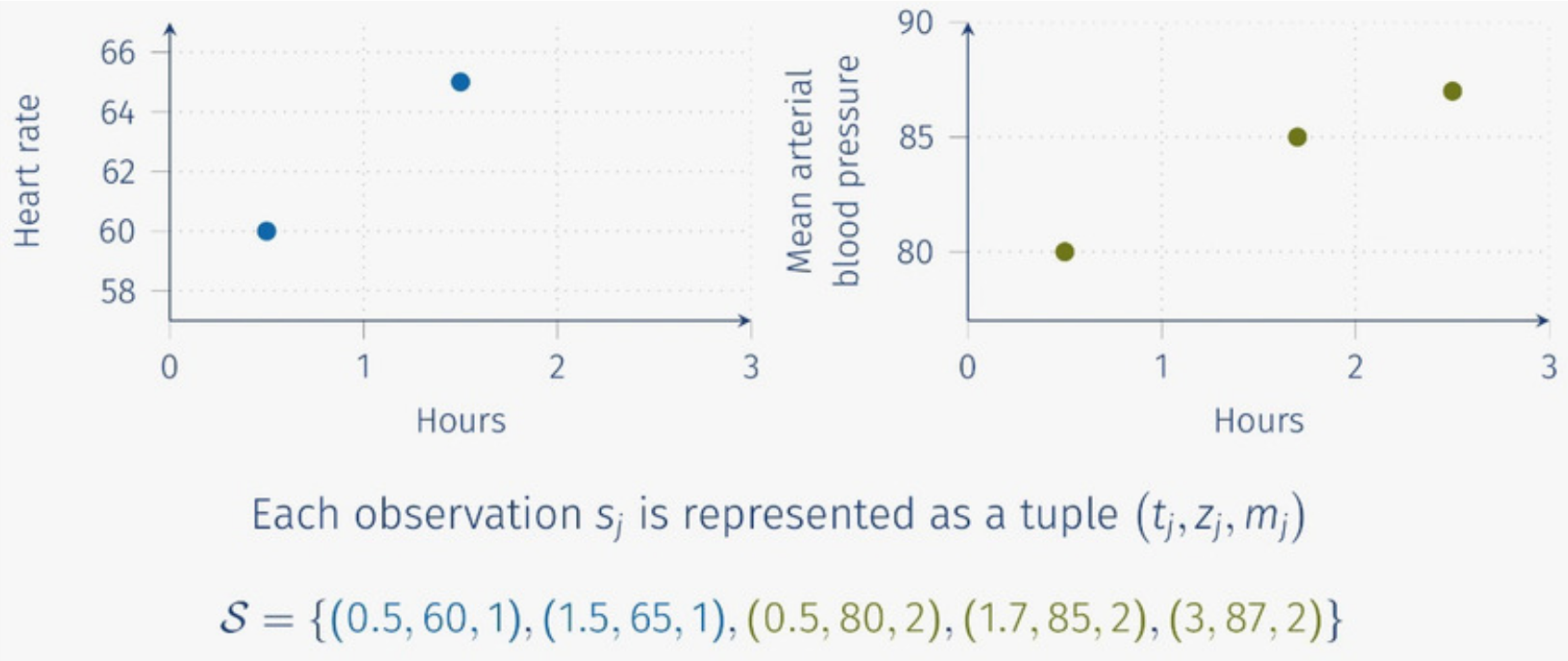
Set Encoding

Embedding, Attention, and Aggregation

Classification

Horn, Max, Michael Moor, Christian Bock, Bastian Rieck, and Karsten Borgwardt. "Set functions for time series." In International Conference on Machine Learning, pp. 4353-4363. PMLR, 2020.
http://proceedings.mlr.press/v119/horn20a/horn20a.pdf & https://slideslive.com/38928275/set-functions-for-time-series

Key Idea in SeFT: Time Series as Set of Observations



Horn, Max, Michael Moor, Christian Bock, Bastian Rieck, and Karsten Borgwardt. "Set functions for time series." In International Conference on Machine Learning, pp. 4353-4363. PMLR, 2020.
http://proceedings.mlr.press/v119/horn20a/horn20a.pdf & https://slideslive.com/38928275/set-functions-for-time-series

Background: *Deep Sets* Framework (Zaheer et. al., NeurIPS 2017)

- Neural network architectures that operate over sets
  - encountered in many applications: compute statistics over sets (e.g. sum of digits in a set of images), classify sets (e.g. LIDAR or RADAR point cloud classification)
- Key requirement:
  - **permutation invariance** if the task is $f : 2^{\mathcal{X}} \to \mathcal{Y}$ (i.e. regression or classification)
  - **permutation equivariance** of the task is $f : \mathcal{X}^M \to \mathcal{Y}^M$ (i.e. transduction)
- Key Results

Theorem 2 *A function $f(X)$ operating on a set X having elements from a countable universe, is a valid set function,* i.e., ***invariant*** *to the permutation of instances in X, iff it can be decomposed in the form $\rho\left(\sum_{x \in X} \phi(x)\right)$, for suitable transformations ϕ and ρ.*

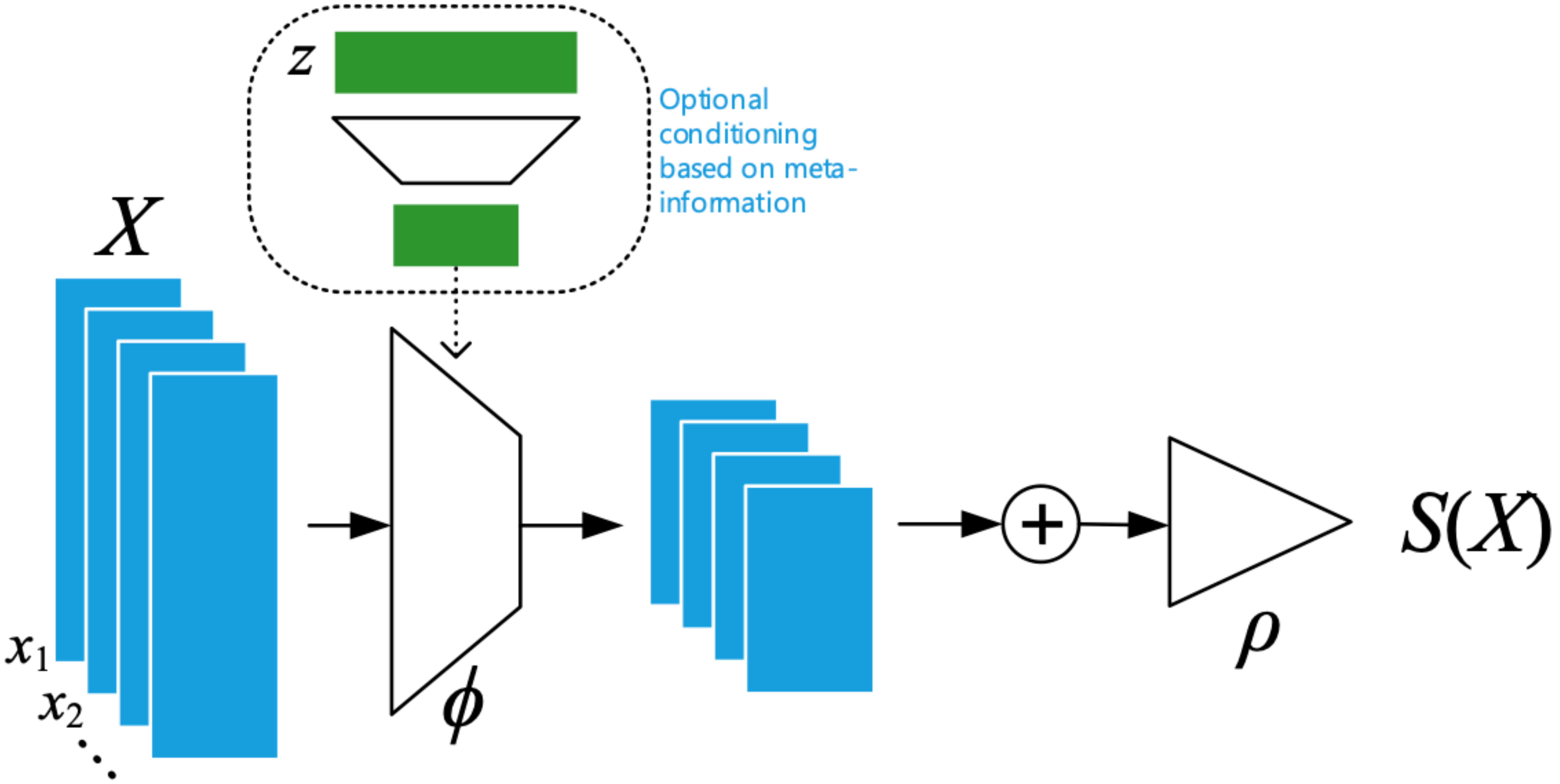
$\mathbf{f}_\Theta(\mathbf{x}) = \boldsymbol{\sigma}(\Theta \mathbf{x})$ is restricted to standard neural network layer

Lemma 3 *The function $\mathbf{f}_\Theta : \mathbb{R}^M \to \mathbb{R}^M$ defined above is permutation **equivariant** iff all the off-diagonal elements of Θ are tied together and all the diagonal elements are equal as well. That is,*

$$\Theta = \lambda \mathbf{I} + \gamma\left(\mathbf{1}\mathbf{1}^\mathsf{T}\right) \qquad \lambda, \gamma \in \mathbb{R} \quad \mathbf{1} = [1, \ldots, 1]^\mathsf{T} \in \mathbb{R}^M \qquad \mathbf{I} \in \mathbb{R}^{M \times M} \textit{ is the identity matrix}$$

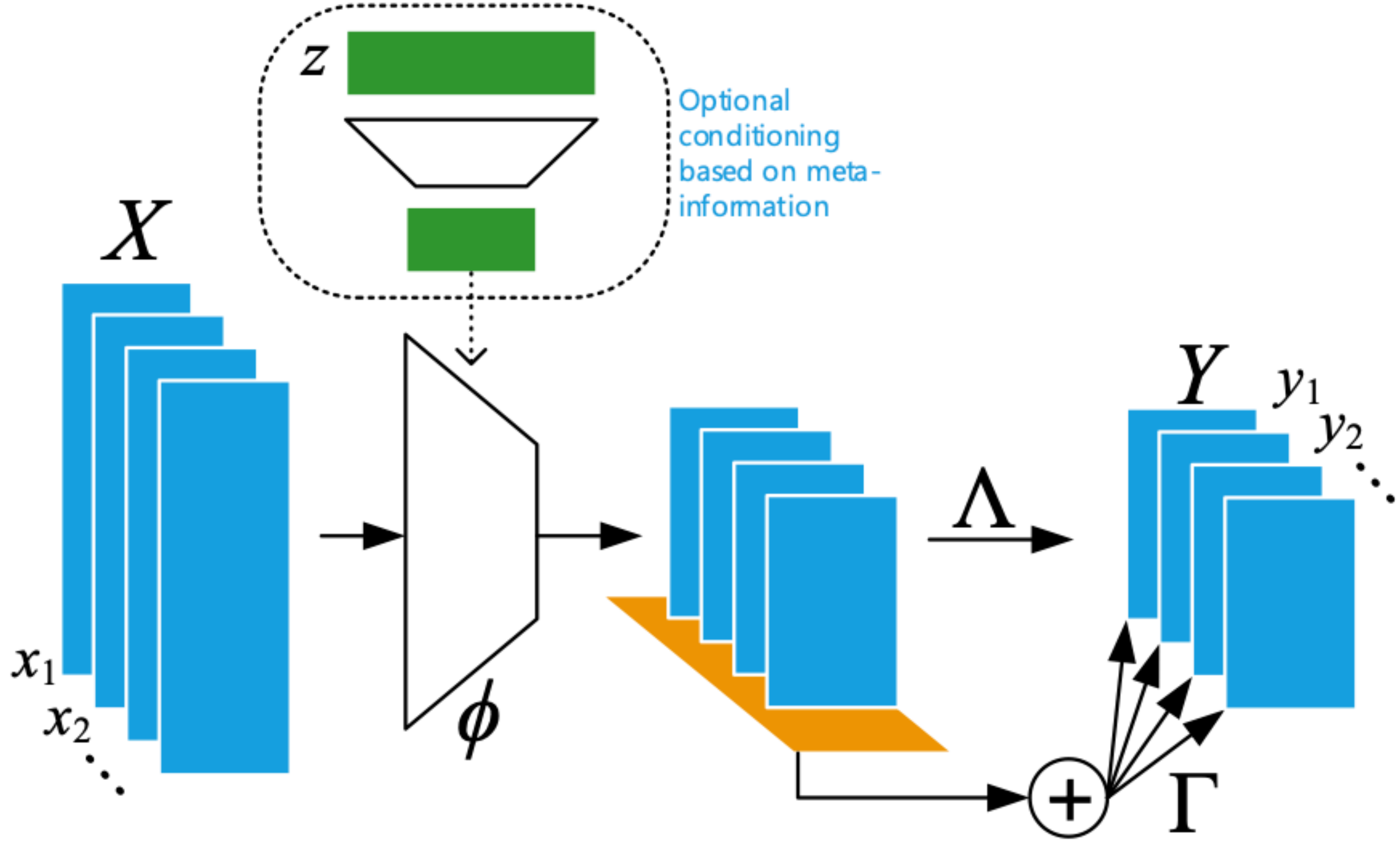
Zaheer, Manzil, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. "Deep sets." arXiv preprint arXiv:1703.06114 (2017).
https://arxiv.org/pdf/1703.06114

Background: Architecture of DeepSets *Invariant* Model



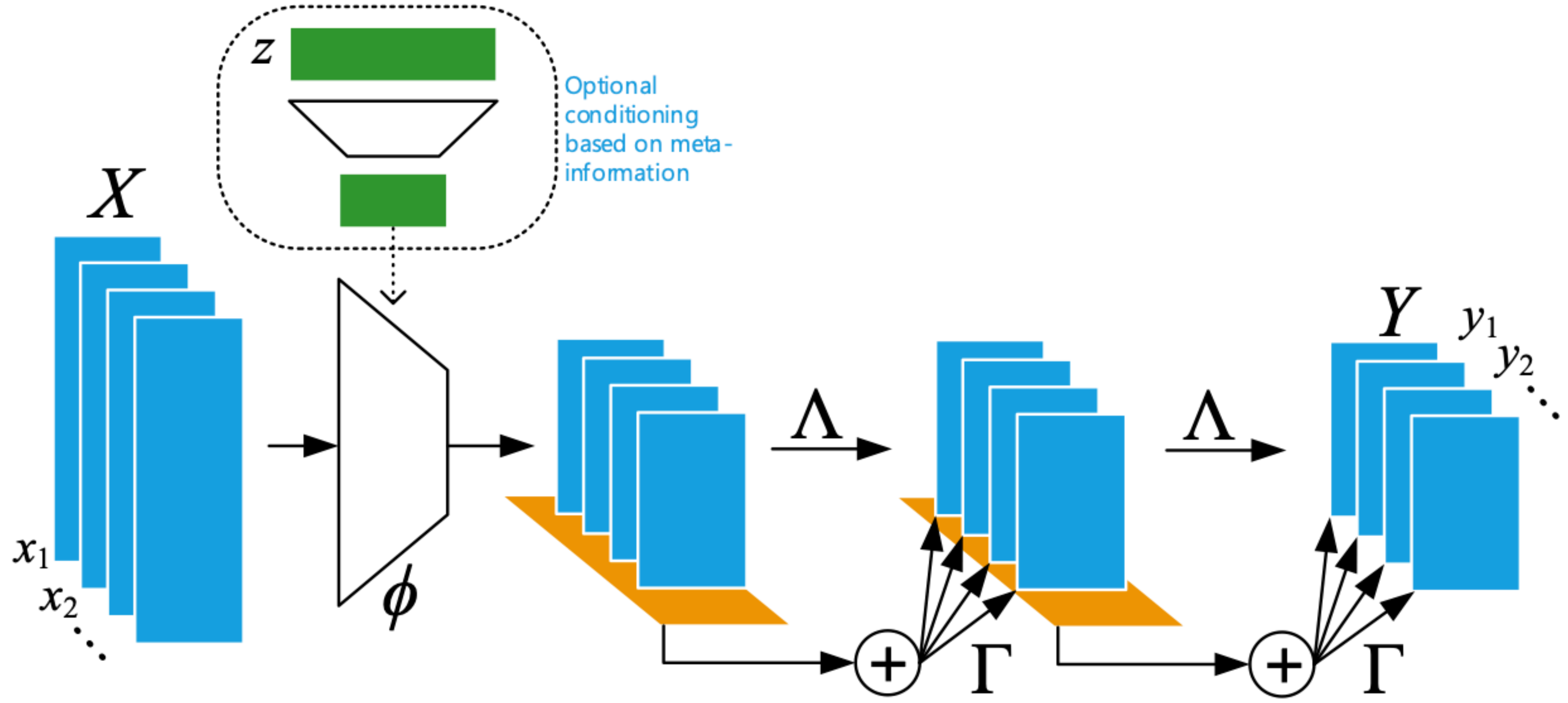
Zaheer, Manzil, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. "Deep sets." arXiv preprint arXiv:1703.06114 (2017).
https://arxiv.org/pdf/1703.06114

Background: Architecture of DeepSets *Equivariant* Model



Zaheer, Manzil, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. "Deep sets." arXiv preprint arXiv:1703.06114 (2017).
https://arxiv.org/pdf/1703.06114

Background: Architecture of DeepSets *Equivariant* Model



Using multiple permutation equivariant layers.

(Since permutation equivariance compose we can stack multiple such layers.)

Zaheer, Manzil, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. "Deep sets." arXiv preprint arXiv:1703.06114 (2017).
https://arxiv.org/pdf/1703.06114

Applying Deep Sets to SeFT

- Sum-decompose the set function to achieve permutation invariance

$$f(\mathcal{S}) = g\left(\frac{1}{|\mathcal{S}|}\sum_{s_j \in \mathcal{S}} h(s_j)\right)$$

where $h: \Omega \rightarrow \mathbb{R}^d$ and $g: \mathbb{R}^d \rightarrow \mathbb{R}^C$ are neural networks

Problem

Influence of an element shrinks as $|\mathcal{S}|$ grows!

Horn, Max, Michael Moor, Christian Bock, Bastian Rieck, and Karsten Borgwardt. "Set functions for time series." In International Conference on Machine Learning, pp. 4353-4363. PMLR, 2020.
http://proceedings.mlr.press/v119/horn20a/horn20a.pdf & https://slideslive.com/38928275/set-functions-for-time-series

Time Encoding in SeFT

- Employs a variant of *positional encoding* seen earlier with transformer

- The time encoding converts the 1-dimensional time axis into a multi-dimensional input by passing the time t of each observation through multiple trigonometric functions of varying frequencies

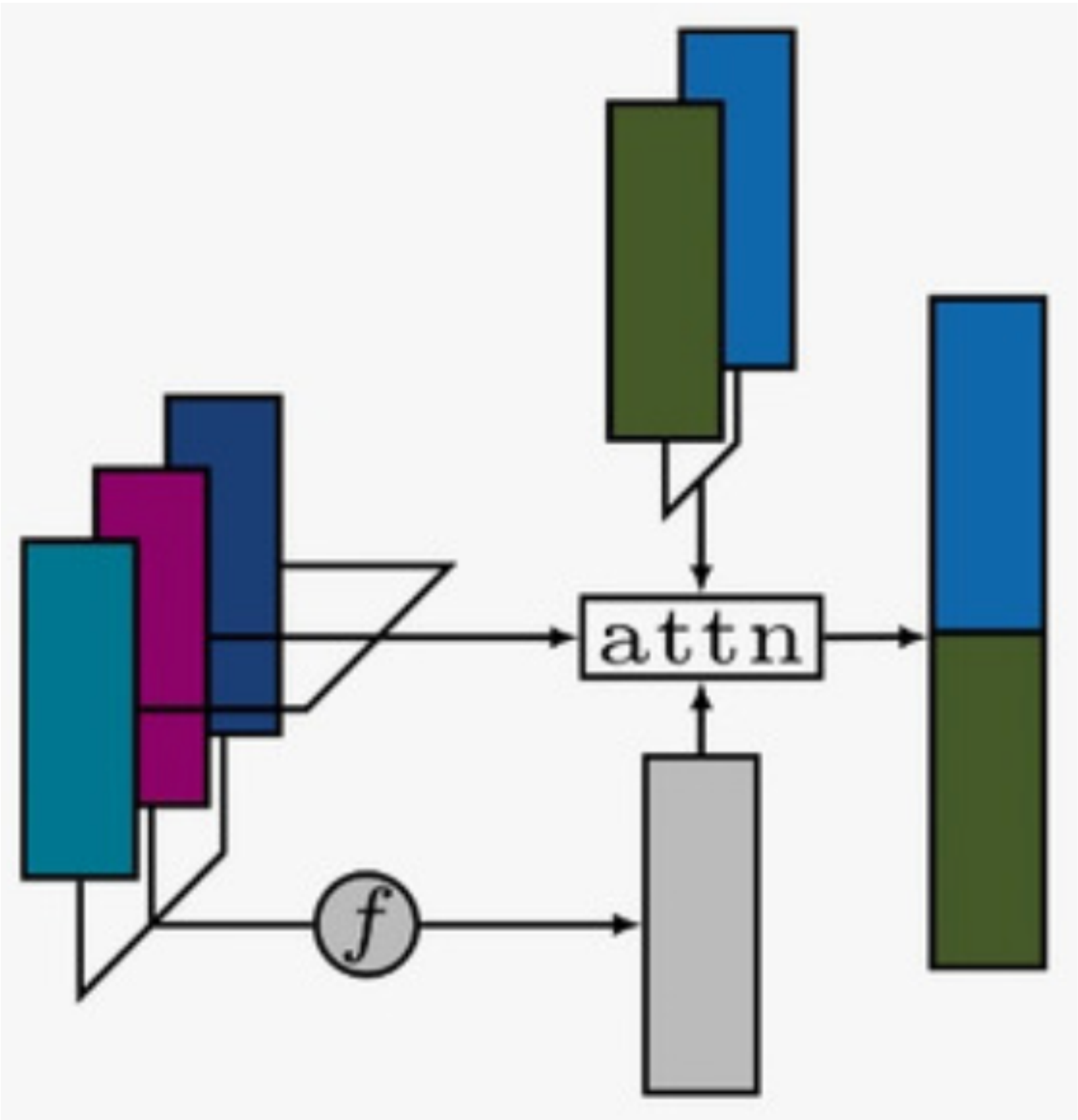
- Given a dimensionality $\tau \in \mathbb{N}^+$ of the time encoding, SeFT encodes the position as $x \in \mathbb{R}^\tau$, where

$$x_{2k}(t) := \sin\left(\frac{t}{t^{2k/\tau}}\right)$$

$$x_{2k+1}(t) := \cos\left(\frac{t}{t^{2k/\tau}}\right) \qquad k \in \{0, \ldots, \tau/2\}$$

Horn, Max, Michael Moor, Christian Bock, Bastian Rieck, and Karsten Borgwardt. "Set functions for time series." In International Conference on Machine Learning, pp. 4353-4363. PMLR, 2020.
http://proceedings.mlr.press/v119/horn20a/horn20a.pdf & https://slideslive.com/38928275/set-functions-for-time-series

Attention-based Aggregation



$$\text{Keys:} \quad K_{j,i} = [f(\mathcal{S}), s_j]^T W_i$$

$$\text{Queries:} \quad Q \in \mathbb{R}^{m \times d}$$

$$\text{Preattentions:} \quad e_{j,i} = \frac{K_{j,i} \cdot Q_i}{\sqrt{d}}$$

$$\text{Attentions:} \quad a_{j,i} = \frac{\exp(e_{j,i})}{\sum_j \exp(e_{j,i})}$$

$$\text{Values:} \quad V_i = \sum_j a_{j,i} h_\theta(s_j)$$

$$\mathcal{L}(\theta, \psi) = \mathbb{E}_{(\mathcal{S},y)\in\mathcal{D}} \left[\ell \left(y; g_\psi \left(\sum_{s_j \in \mathcal{S}} a(\mathcal{S}, s_j) h_\theta(s_j) \right) \right) \right]$$

Horn, Max, Michael Moor, Christian Bock, Bastian Rieck, and Karsten Borgwardt. "Set functions for time series." In International Conference on Machine Learning, pp. 4353-4363. PMLR, 2020.
http://proceedings.mlr.press/v119/horn20a/horn20a.pdf & https://slideslive.com/38928275/set-functions-for-time-series

SeFT Evaluation

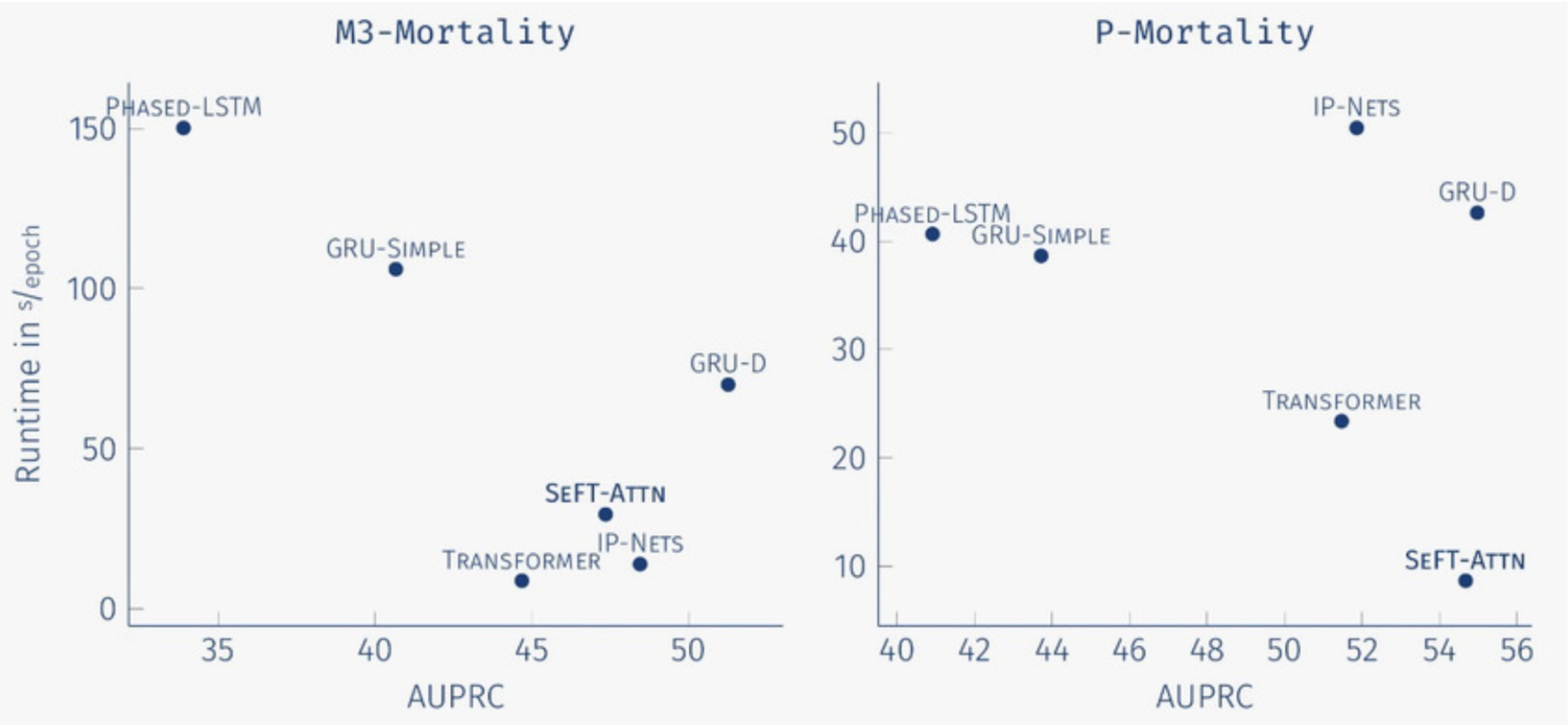
Dataset	Model	Accuracy	AUPRC	AUROC	s/epoch
M3M	GRU-D	77.0 ± 1.5	**52.0 ± 0.8**	**85.7 ± 0.2**	133 ± 8
	GRU-SIMPLE	78.1 ± 1.3	43.6 ± 0.4	82.8 ± 0.0	140 ± 7
	IP-NETS	78.3 ± 0.7	*48.3 ± 0.4*	83.2 ± 0.5	81.2 ± 8.5
	PHASED-LSTM	73.8 ± 3.3	37.1 ± 0.5	80.3 ± 0.4	166 ± 7
	TRANSFORMER	*77.4 ± 5.6*	42.6 ± 1.0	82.1 ± 0.3	*20.1 ± 0.1*
	LATENT-ODE†	72.8 ± 1.7	39.5 ± 0.5	80.9 ± 0.2	4622
	SEFT-ATTN	**79.0 ± 2.2**	46.3 ± 0.5	*83.9 ± 0.4*	**14.5 ± 0.5**
P12	GRU-D	80.0 ± 2.9	*53.7 ± 0.9*	**86.3 ± 0.3**	8.67 ± 0.49
	GRU-SIMPLE	*82.2 ± 0.2*	42.2 ± 0.6	80.8 ± 1.1	30.0 ± 2.5
	IP-NETS	79.4 ± 0.3	51.0 ± 0.6	*86.0 ± 0.2*	25.3 ± 1.8
	PHASED-LSTM	76.8 ± 5.2	38.7 ± 1.5	79.0 ± 1.0	44.6 ± 2.3
	TRANSFORMER	**83.7 ± 3.5**	**52.8 ± 2.2**	**86.3 ± 0.8**	**6.06 ± 0.06**
	LATENT-ODE†	76.0 ± 0.1	50.7 ± 1.7	85.7 ± 0.6	3500
	SEFT-ATTN	75.3 ± 3.5	52.4 ± 1.1	85.1 ± 0.4	*7.62 ± 0.10*

Horn, Max, Michael Moor, Christian Bock, Bastian Rieck, and Karsten Borgwardt. "Set functions for time series." In International Conference on Machine Learning, pp. 4353-4363. PMLR, 2020.
http://proceedings.mlr.press/v119/horn20a/horn20a.pdf & https://slideslive.com/38928275/set-functions-for-time-series

SeFT Result: Performance vs. Runtime

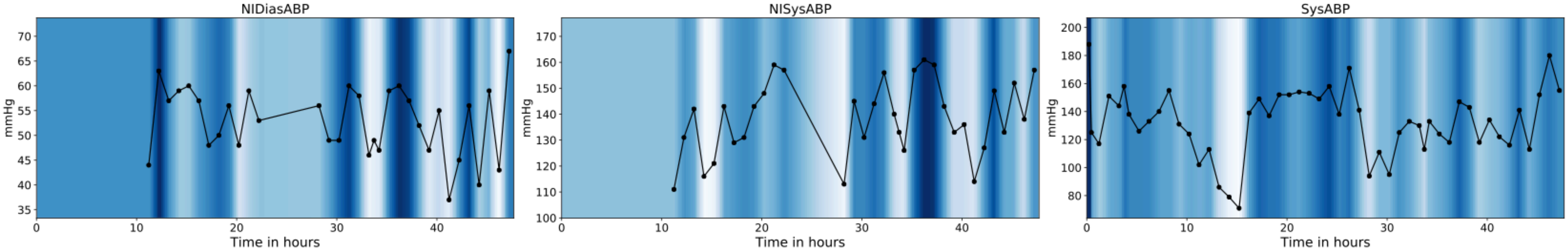
MIMIC-III Mortality Prediction Task

Physionet 2012 Mortality Prediction Task



Horn, Max, Michael Moor, Christian Bock, Bastian Rieck, and Karsten Borgwardt. "Set functions for time series." In International Conference on Machine Learning, pp. 4353-4363. PMLR, 2020.
http://proceedings.mlr.press/v119/horn20a/horn20a.pdf & https://slideslive.com/38928275/set-functions-for-time-series

SeFT's Outputs are Explainable

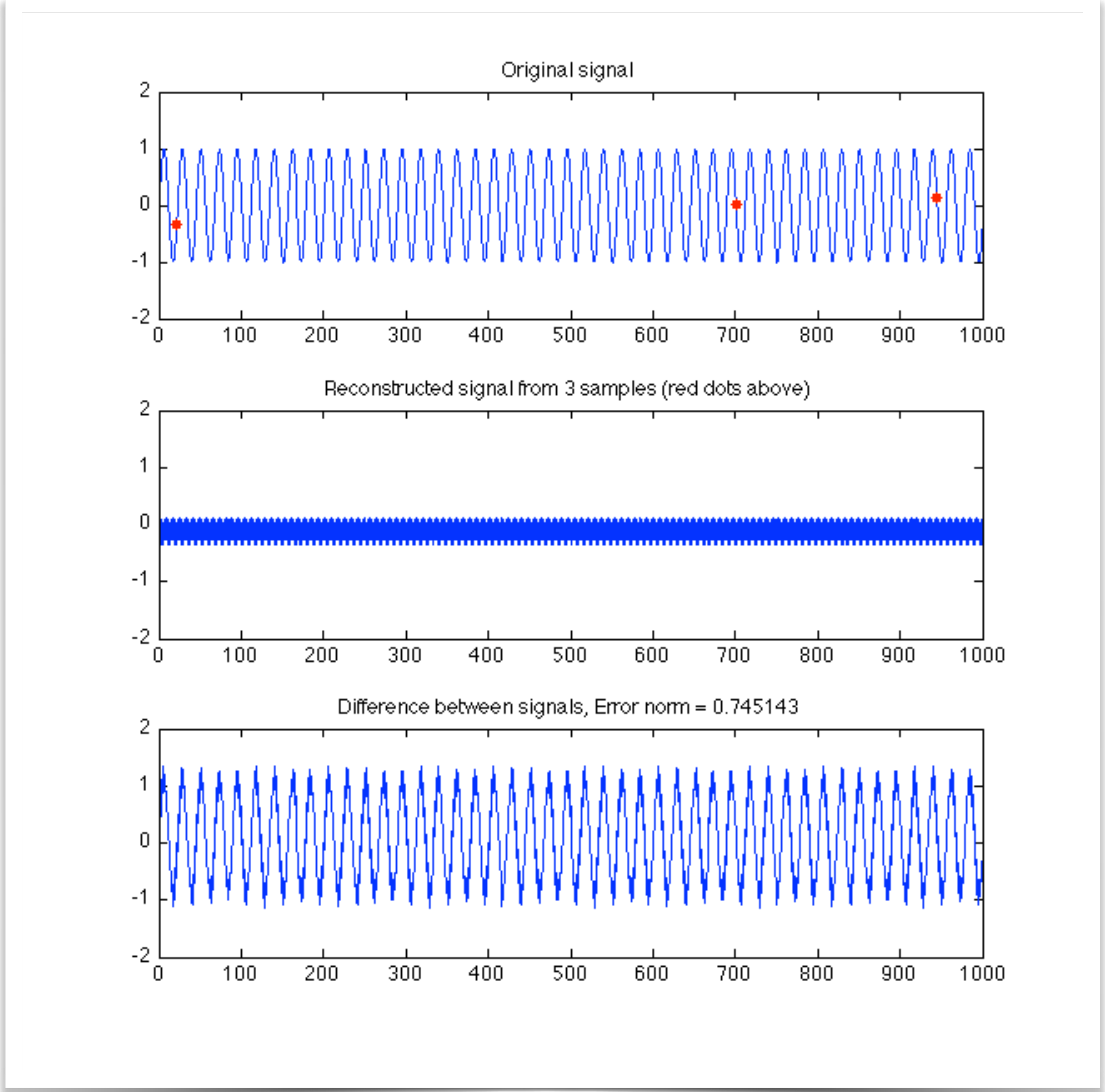


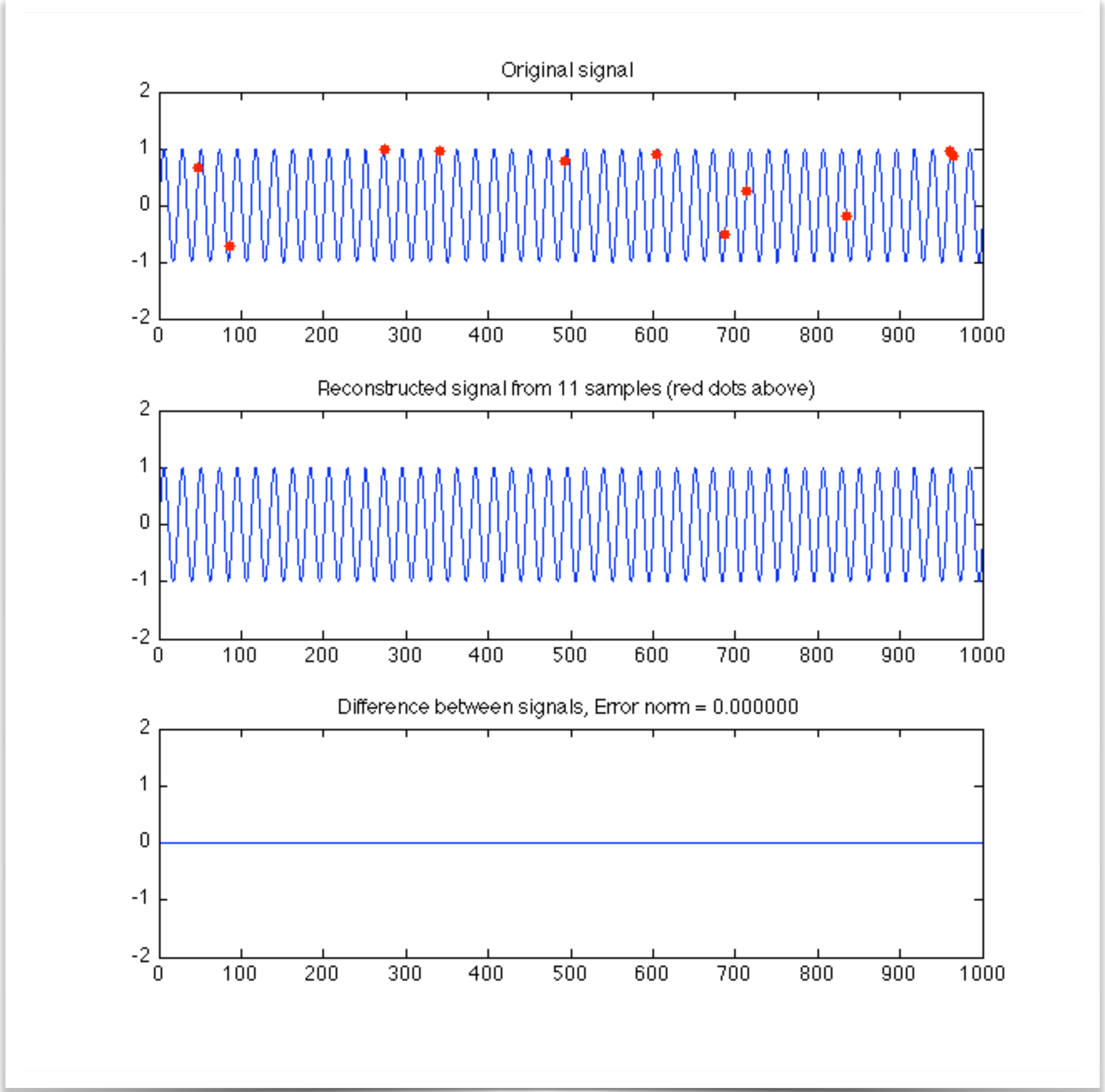
Allows a per-observation quantification of importance

Horn, Max, Michael Moor, Christian Bock, Bastian Rieck, and Karsten Borgwardt. "Set functions for time series." In International Conference on Machine Learning, pp. 4353-4363. PMLR, 2020.
http://proceedings.mlr.press/v119/horn20a/horn20a.pdf & https://slideslive.com/38928275/set-functions-for-time-series

Combining Compressed Sensing and Machine Learning

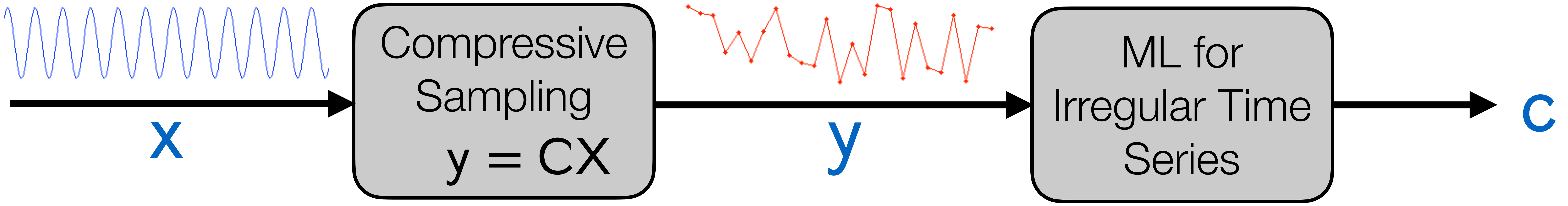
Recall: Compressed Sensing





Combining Compressive Sensing and Machine Learning

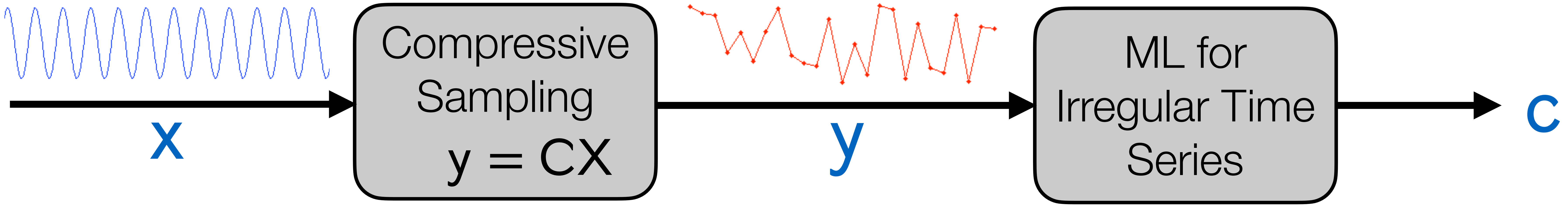
Option 1



Does not work well as |y| ≪ |x|

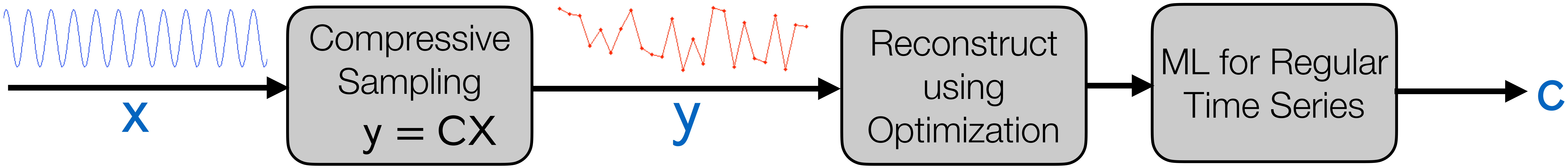
Combining Compressive Sensing and Machine Learning

Option 1



Does not work well as $|y| \ll |x|$

Option 2



Costly!

$\text{argmin}_s \ ||s||_1 \text{ , s.t. } ||C\Psi s - y||_2 < \epsilon$

Could we Combine Compressive Sensing and Deep Learning?

Compressed Learning with Deep Neural Networks

Adler, Amir, Michael Elad, and Michael Zibulevsky. "Compressed learning: A deep neural network approach." arXiv preprint arXiv:1610.09615 (2016).
https://arxiv.org/pdf/1610.09615

Compressed Learning with Deep Neural Networks

- Compressed Learning: direct inference from compressive measurements is feasible with high classification accuracies
  - Calderbank, Jafarpour, and Schapire. "Compressed learning: Universal sparse dimensionality reduction and learning in the measurement domain." preprint (2009).
  - Proved that under certain conditions the performance of a linear SVM classifier operating in the compressed sensing domain $y = Cx$ is almost equivalent to the performance of the best linear threshold classifier operating in the signal domain x

Adler, Amir, Michael Elad, and Michael Zibulevsky. "Compressed learning: A deep neural network approach." arXiv preprint arXiv:1610.09615 (2016).
https://arxiv.org/pdf/1610.09615

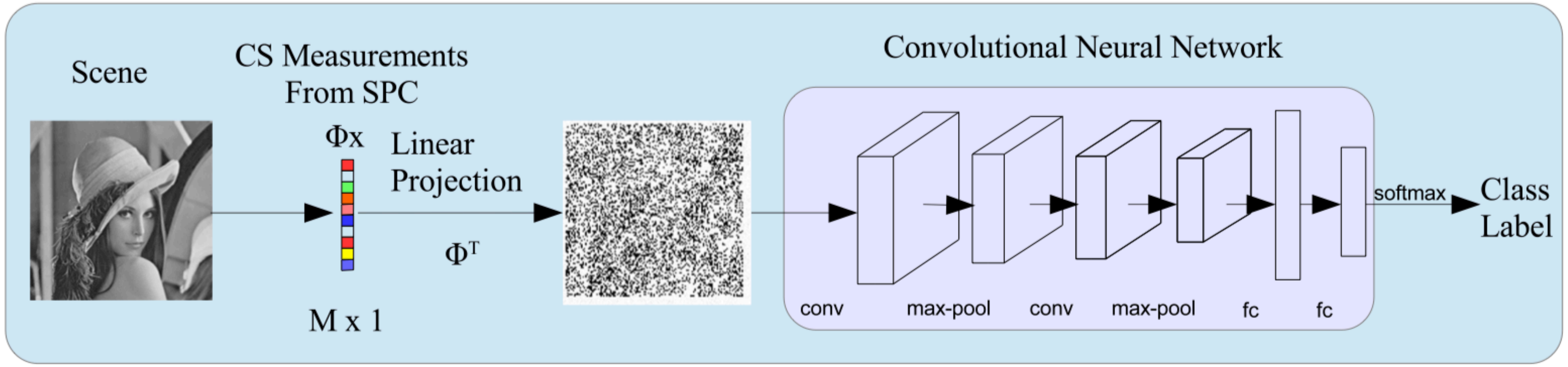
Compressed Learning with Deep Neural Networks

- Compressed Learning: direct inference from compressive measurements is feasible with high classification accuracies
  - Calderbank, Jafarpour, and Schapire. "Compressed learning: Universal sparse dimensionality reduction and learning in the measurement domain." preprint (2009).
  - Proved that under certain conditions the performance of a linear SVM classifier operating in the compressed sensing domain $y = Cx$ is almost equivalent to the performance of the best linear threshold classifier operating in the signal domain x

- Subsequent work combined CL with DNNs: projected measurement vector $z = C^T y$ (same shape as x) as the input to a convolutional neural network
  - Lohit, Kulkarni, and Turaga. "Direct inference on compressive measurements using convolutional neural networks." IEEE Intl Conf on Image Proc. (ICIP), pp. 1913-1917.
  - Showed pretty good results on MNIST and ImageNet classification by training on projected measurement $z = C^T y$ instead of on the original signal x

Adler, Amir, Michael Elad, and Michael Zibulevsky. "Compressed learning: A deep neural network approach." arXiv preprint arXiv:1610.09615 (2016).
https://arxiv.org/pdf/1610.09615

Direct Inference on Compressive Measurements using CNN



Note: specific CNN architecture is for illustration only - specific architecture will depend on application.

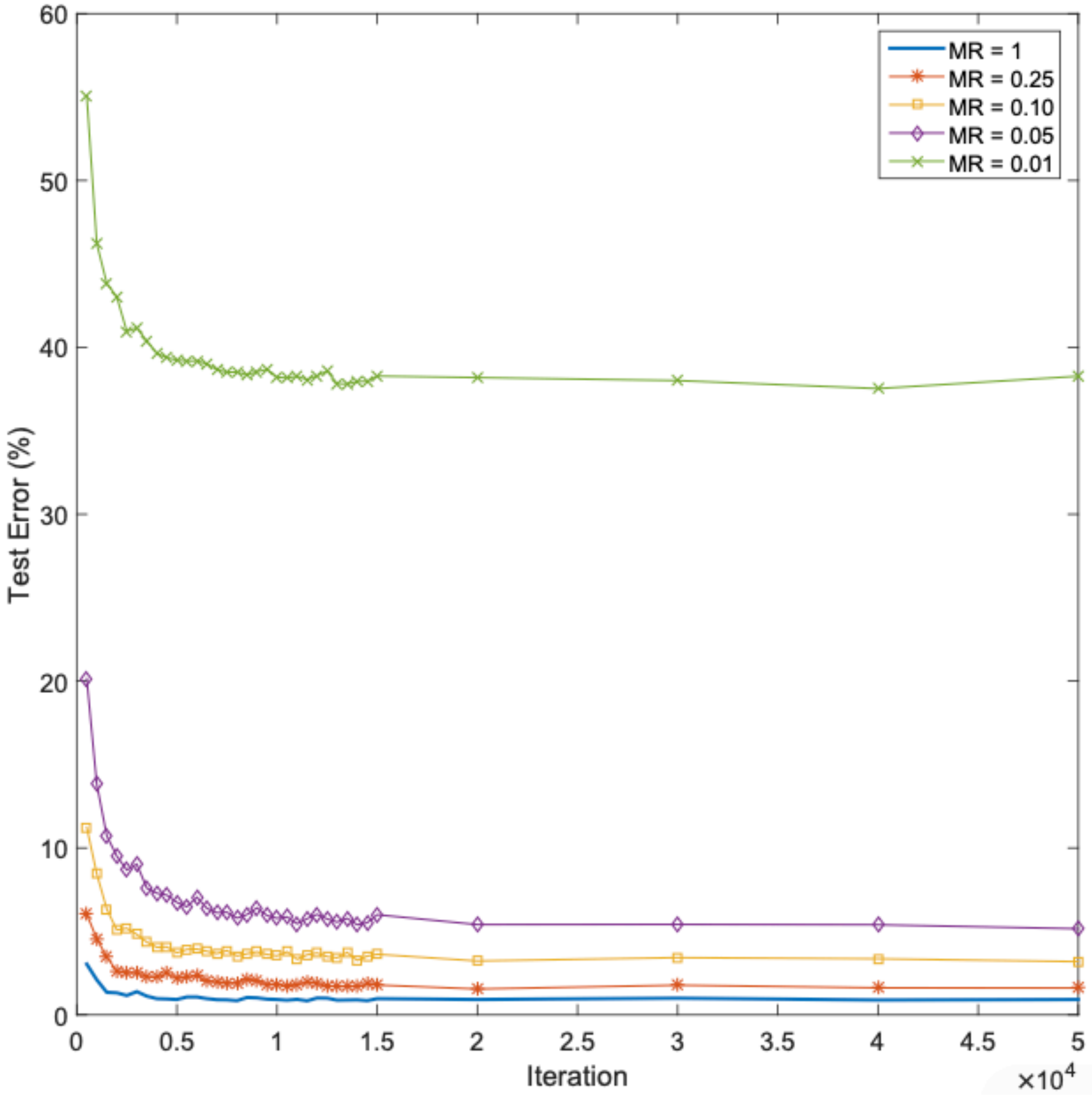
Architecture

Measurement Rate (MR)	Number of Measurements	Test Error	
		Smashed Filters [4]	Our Method
1 (Oracle)	784	13.86%	0.89%
0.25	196	27.42%	1.63%
0.10	78	43.55%	2.99%
0.05	39	53.21%	5.18%
0.01	8	63.03%	41.06%

MNIST Classification

Measurement Rate	No. of Measurements	Accuracy
1 (Oracle)	65536	56.88%
0.25	16384	39.22%
0.10	6554	29.84%

ImageNet Classification

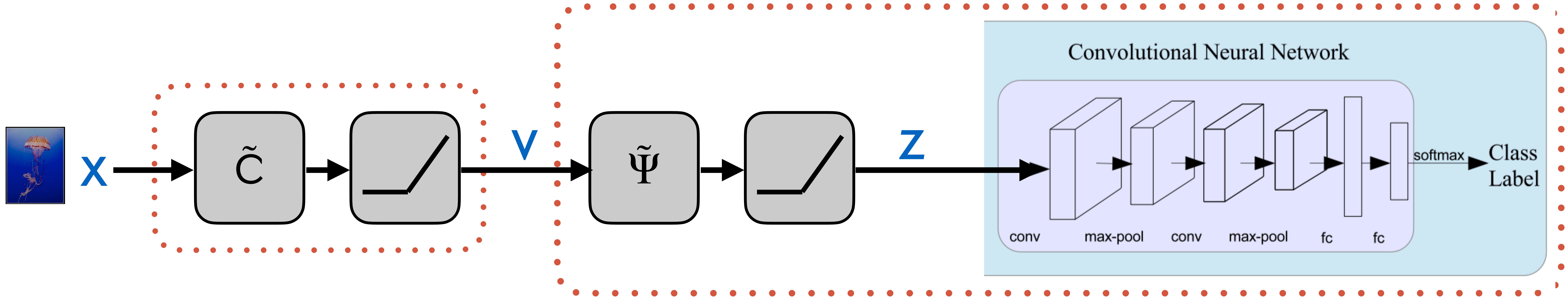


Convergence of Test Error

Lohit, Suhas, Kuldeep Kulkarni, and Pavan Turaga. "Direct inference on compressive measurements using convolutional neural networks." In *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 1913-1917. IEEE, 2016.
https://ieeexplore.ieee.org/iel7/7527113/7532277/07532691.pdf

End-to-end Deep Learning Solution for Compressed Learning

- Idea: jointly optimize the sensing matrix C and the inference operator (i.e. the CNN)
  - Adler, Amir, Michael Elad, and Michael Zibulevsky. "Compressed learning: A deep neural network approach." arXiv preprint arXiv:1610.09615 (2016).
- Approach
  - The first layer learns and performs the sensing matrix C
  - The subsequent layers (a fully-connected layer followed by a CNN) perform the non-linear inference stage
  - The second fully-connected layer performs operation similar to $z = C^T y$ but a different matrix $\tilde{C}$ is learnt
  - The first and second layers are followed by ReLU
  - The two components of end-to-end CL detached after training



https://arxiv.org/pdf/1610.09615

End-to-end Deep Learning Solution for Compressed Learning

- Idea: jointly optimize the sensing matrix $\mathrm{C}$and the inference operator (i.e. the CNN)
  - Adler, Amir, Michael Elad, and Michael Zibulevsky. "Compressed learning: A deep neural network approach." arXiv preprint arXiv:1610.09615 (2016).
- Approach
  - The first layer learns and performs the sensing matrix C
  - The subsequent layers (a fully-connected layer followed by a CNN) perform the non-linear inference stage
  - The second fully-connected layer performs operation similar to $\mathrm{z} = \mathrm{C}^\mathrm{T}\mathrm{y}$ but a different matrix $\tilde{\mathrm{C}}$ is learnt
  - The first and second layers are followed by ReLU
  - The two components of end-to-end CL detached after training

Sensing Rate	No. of Measurements	Smashed Filters [12]	Random Sensing + CNN [4]	Proposed
0.25	196	27.42%	1.63%	**1.56%**
0.1	78	43.55%	2.99%	**1.91%**
0.05	39	53.21%	5.18%	**2.86%**
0.01	8	63.03%	41.06%	**6.46%**

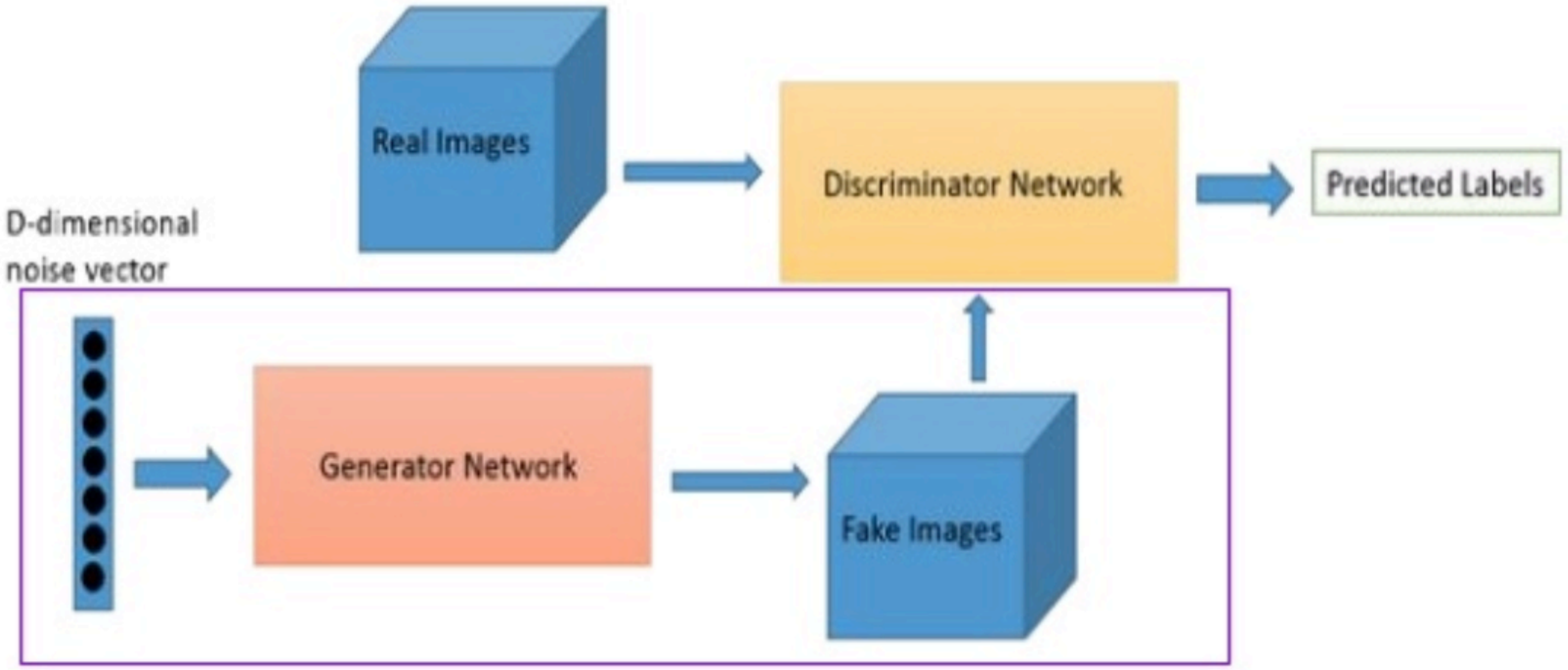
Classification Error (%) for the MNIST handwritten digits dataset vs. sensing rate R = M/N (averaged over 10,000 test images)

https://arxiv.org/pdf/1610.09615

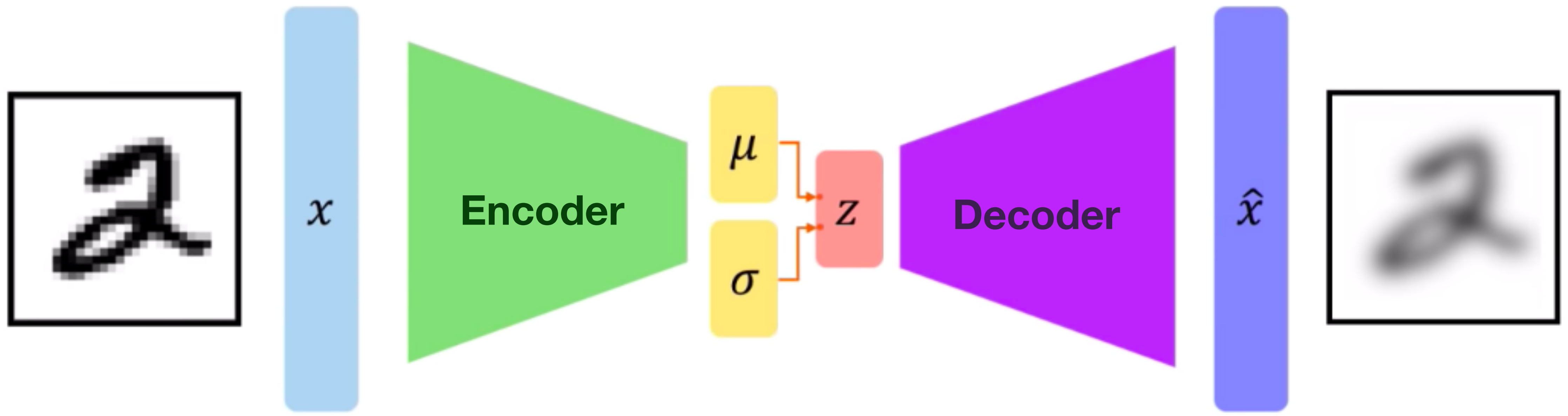
Compressed Sensing Using Generative Models

- CS compresses by taking random linear projections (measurement matrix) of the original signal, and reconstructs by exploiting sparsity present in “natural” signals

- Instead of relying on sparsity, one can use *structure* from a generative model.
  - GANs and VAEs



Generative Adversarial Networks



Variational Auto Encoder

Bora, Ashish, Ajil Jalal, Eric Price, and Alexandros G. Dimakis. "Compressed sensing using generative models." In International Conference on Machine Learning, pp. 537-546. PMLR, 2017.
http://proceedings.mlr.press/v70/bora17a/bora17a.pdf

Compressed Sensing Using Generative Models

- CS compresses by taking random linear projections (measurement matrix) of the original signal, and reconstructs by exploiting sparsity present in “natural” signals

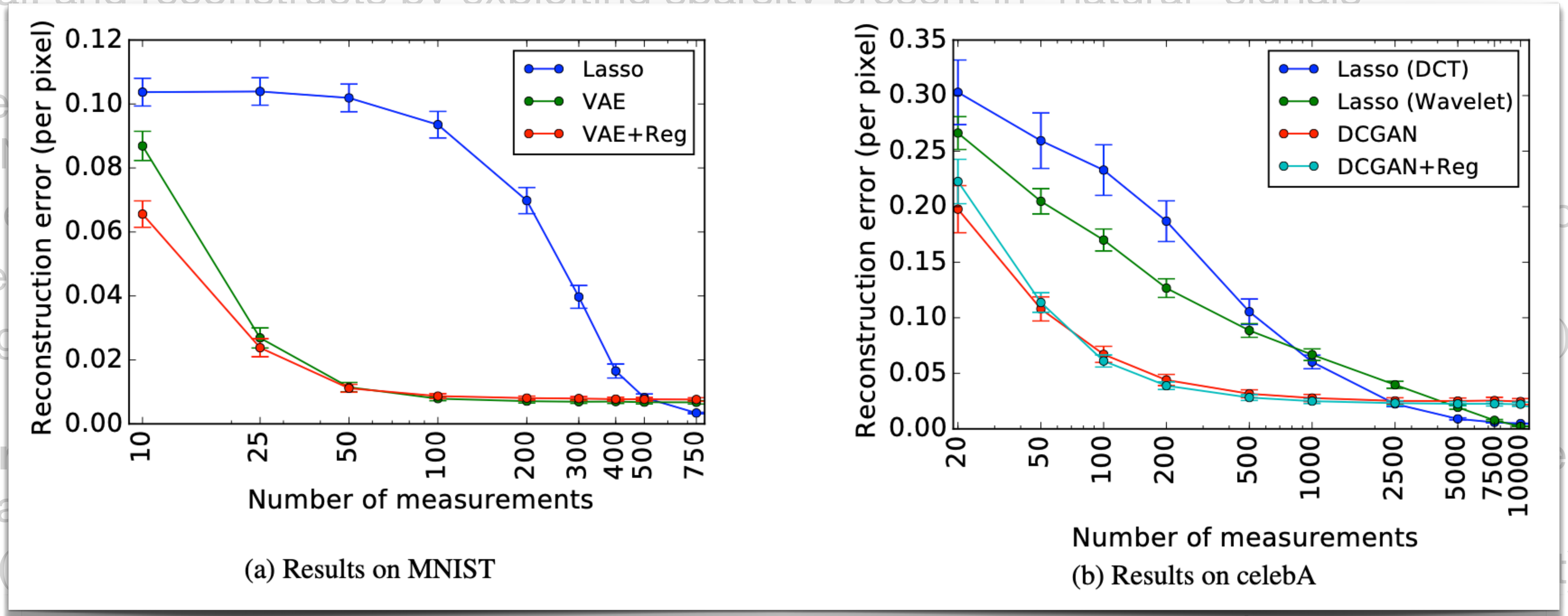
- Instead of relying on sparsity, one can use *structure* from a generative model.
  - GANs and VAEs
  - A generative model is given by a deterministic function $G : \mathbb{R}^k \to \mathbb{R}^n$, and a distribution P_Z over $z \in \mathbb{R}^k$.
  - To generate a sample from the generator, we draw $z \sim P_Z$ and the sample then is $G(z)$

- **Approach:** find a vector in representation space s.t. the corresponding vector in the sample space matches the observed measurements, i.e. optimize $loss(z) = ||AG(z) - y||_2^2$ (highly non-convex, approximated using gradient descent)
  - If the optimization procedure gives $\hat{z}$, then reconstruct by computing $\hat{x} = G(\hat{z})$

Bora, Ashish, Ajil Jalal, Eric Price, and Alexandros G. Dimakis. "Compressed sensing using generative models." In International Conference on Machine Learning, pp. 537-546. PMLR, 2017.
http://proceedings.mlr.press/v70/bora17a/bora17a.pdf

Compressed Sensing Using Generative Models

- CS compresses by taking random linear projections (measurement matrix) of the original signal, and reconstructs by exploiting sparsity present in "natural" signals



(a) Results on MNIST

(b) Results on celebA

- If the optimization procedure gives $\hat{z}$, then reconstruct by computing $\hat{x} = G(\hat{z})$

Bora, Ashish, Ajil Jalal, Eric Price, and Alexandros G. Dimakis. "Compressed sensing using generative models." In International Conference on Machine Learning, pp. 537-546. PMLR, 2017.
http://proceedings.mlr.press/v70/bora17a/bora17a.pdf

References

- Zaheer, Manzil, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. "Deep sets." arXiv preprint arXiv:1703.06114 (2017).
- Chen, Ricky TQ, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. "Neural ordinary differential equations." arXiv preprint arXiv:1806.07366 (2018).
- Shukla, Satya Narayan, and Benjamin M. Marlin. "Interpolation-prediction networks for irregularly sampled time series." arXiv preprint arXiv:1909.07782 (2019).
- Rubanova, Yulia, Ricky TQ Chen, and David Duvenaud. "Latent ODEs for irregularly-sampled time series." arXiv preprint arXiv:1907.03907 (2019).
- Horn, Max, Michael Moor, Christian Bock, Bastian Rieck, and Karsten Borgwardt. "Set functions for time series." In International Conference on Machine Learning, pp. 4353-4363. PMLR, 2020.
- Shukla, Satya Narayan, and Benjamin M. Marlin. "A Survey on Principles, Models and Methods for Learning from Irregularly Sampled Time Series: From Discretization to Attention and Invariance." arXiv preprint arXiv:2012.00168 (2020).
- Shukla, Satya Narayan, and Benjamin M. Marlin. "Multi-Time Attention Networks for Irregularly Sampled Time Series." arXiv preprint arXiv:2101.10318 (2021).

The End