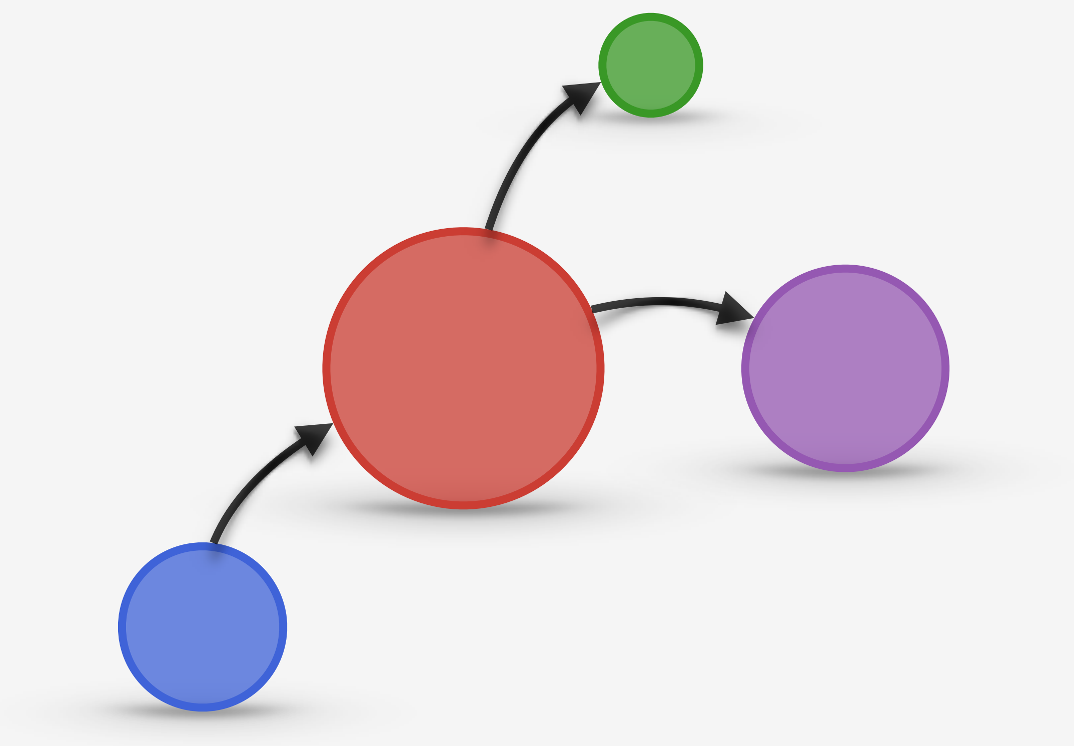
NetworkHawkesProcesses.jl

Networks + Hawkes Processes + Julia



Colin Swaney
Data-Driven Social Science Initiative
@ Princeton University

DATA-DRIVEN SOCIAL SCIENCE INITIATIVE

What is NetworkHawkesProcesses.jl?

Overview

NetworkHawkesProcesses.jl is a **pure Julia framework for defining, simulating, and performing inference** on a class of probabilistic models that permit **simultaneous inference on the structure of a network and its event generating process**—the network Hawkes processes (Linderman, 2016). The event generating process is assumed to follow an **auto-regressive, multivariate Poisson process** known as a Hawkes process. Connections between nodes—the network "structure"—are assumed to follow any standard network model (i.e., independent connections). Combining these models provides a disciplined method for **discovering latent network structure from event data** observed in neuroscience, finance, and beyond.

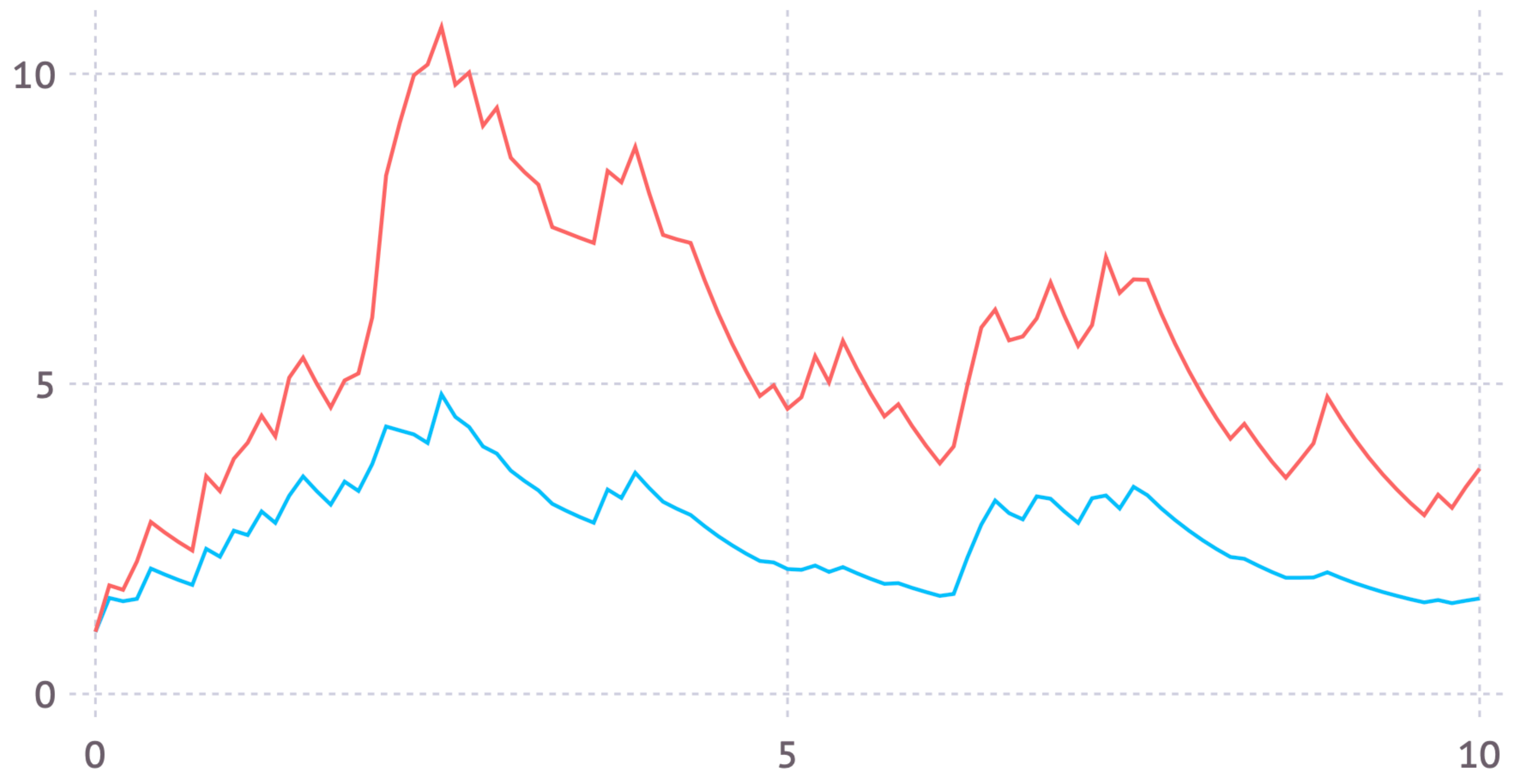
Package Features
Overview

- Uses modular design to support extensible components
- Supports continuous and discrete models
- Implements simulation via Poisson thinning
- Provides multiple inference methods for all models
- Accelerates estimation via Julia's built-in multithreading module

Hawkes Processes

Basics

$$\lambda_n(t \mid \mathcal{H}_t) = \lambda_n^{(0)} + \sum_{m=1}^{M} h_{c_m \to n}(t - s_m)$$



DATA-DRIVEN SOCIAL SCIENCE INITIATIVE

Hawkes Processes
Baselines

- Homogeneous

$$\lambda^{(0)}(t) = const.$$

- Log-Gaussian Cox

$$\begin{aligned} \mathbf{f} &= (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N)) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \\ \mu_i &= m(\mathbf{x}_i) \\ \Sigma_{i,j} &= K(\mathbf{x}_i, \mathbf{x}_j) \end{aligned}$$

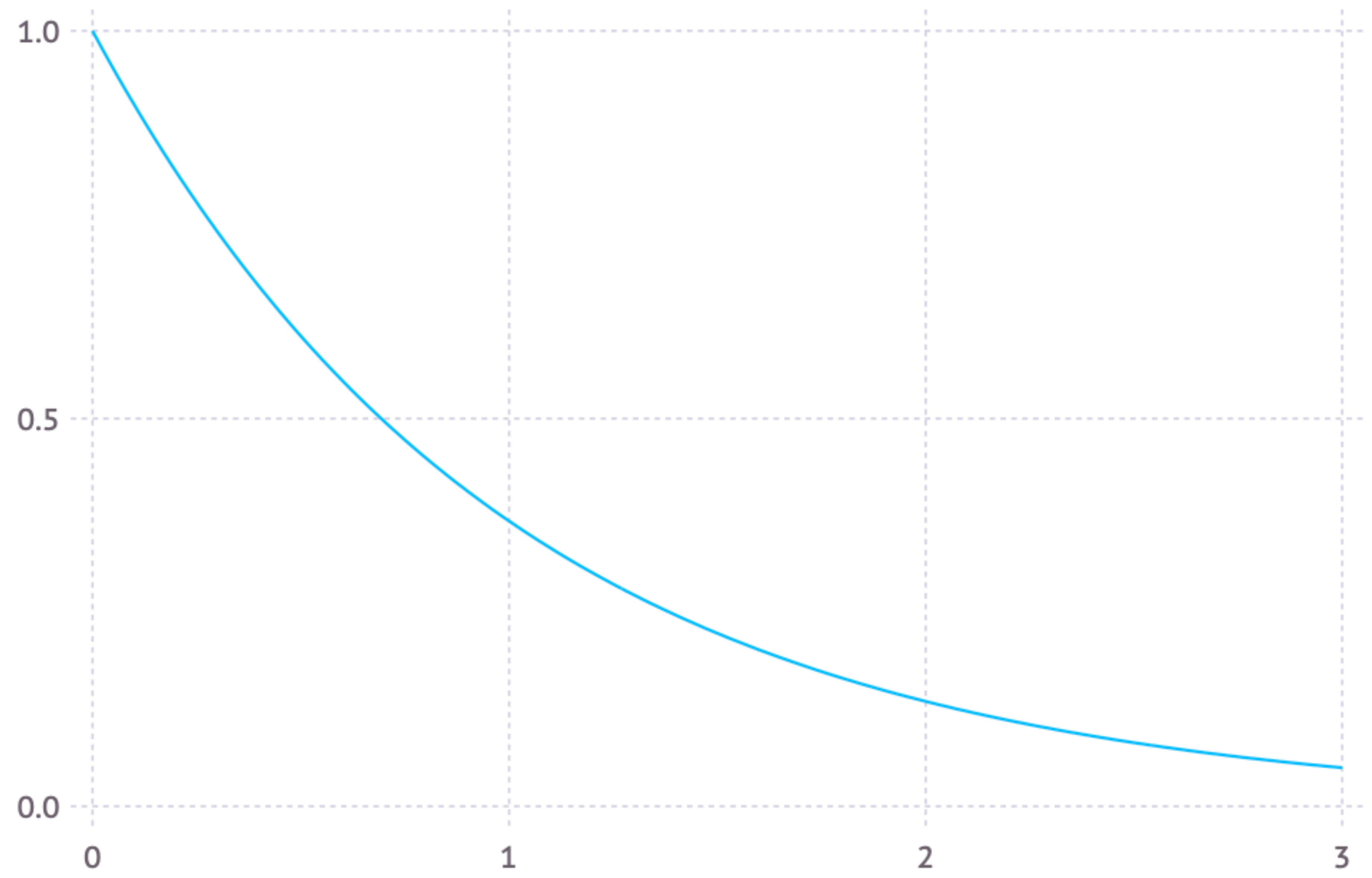
$$\begin{aligned} y &\sim \mathcal{GP}(0, K) \\ \lambda^{(0)}(t) &= \exp\{\mu + y(t)\} \end{aligned}$$

Hawkes Processes

Impulse Responses

- Exponential

$$\hbar(\Delta t; \theta_{n \to n'}) = \theta_{n \to n'} \exp\{-\theta_{n \to n'} \Delta t\}$$

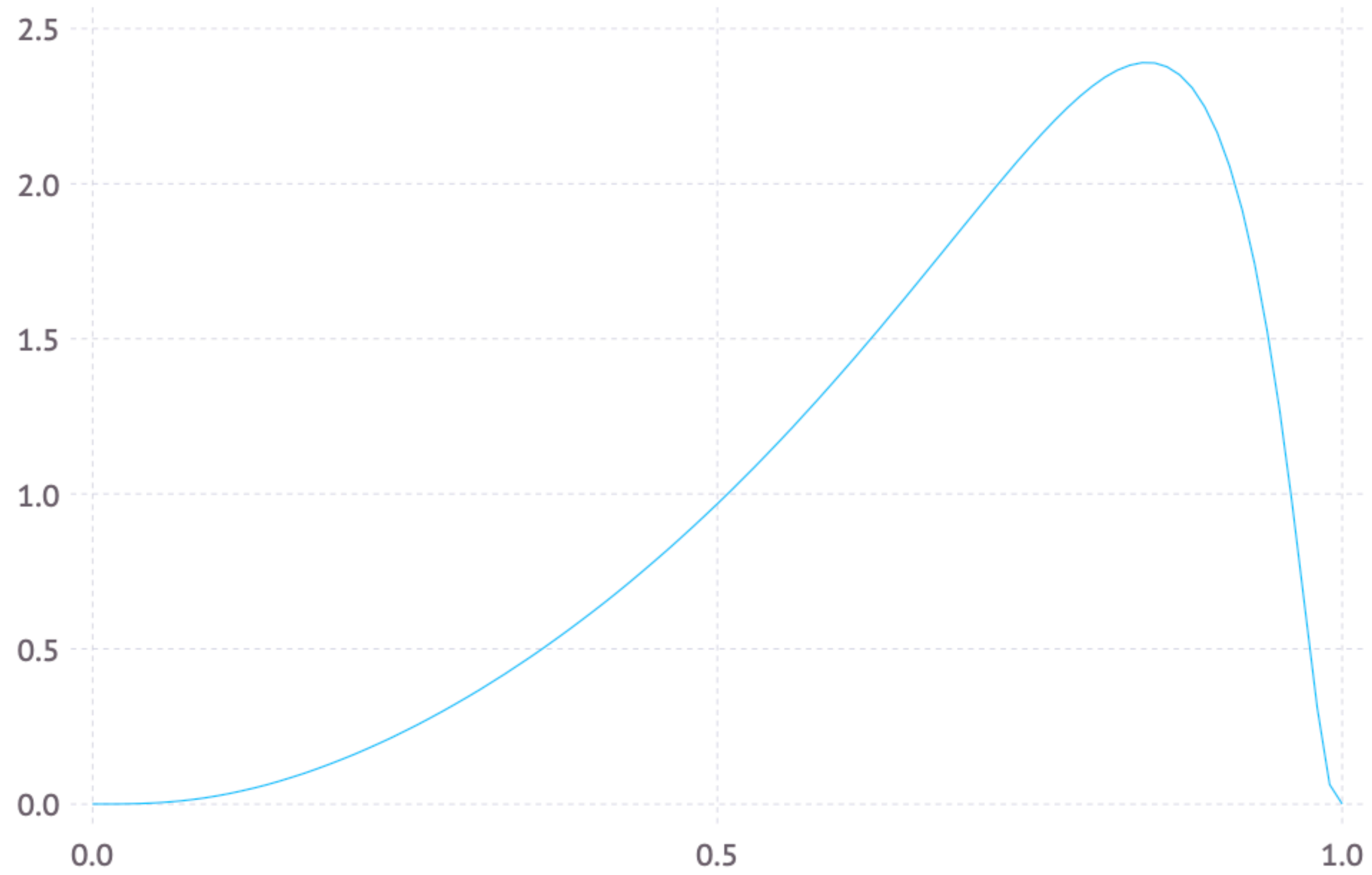


DATA-DRIVEN SOCIAL SCIENCE INITIATIVE

Hawkes Processes
Impulse Responses

- Logistic-Normal

$$\hbar(\Delta t;\, \mu_{n\to n'}, \tau_{n\to n'}) = \frac{1}{Z} \exp\left\{ \frac{-\tau_{n\to n'}}{2} \left(\sigma^{-1}\left(\frac{\Delta t}{\Delta t_{\mathsf{max}}} \right) - \mu_{n\to n'} \right)^2 \right\}$$



DATA-DRIVEN SOCIAL SCIENCE INITIATIVE

Hawkes Processes

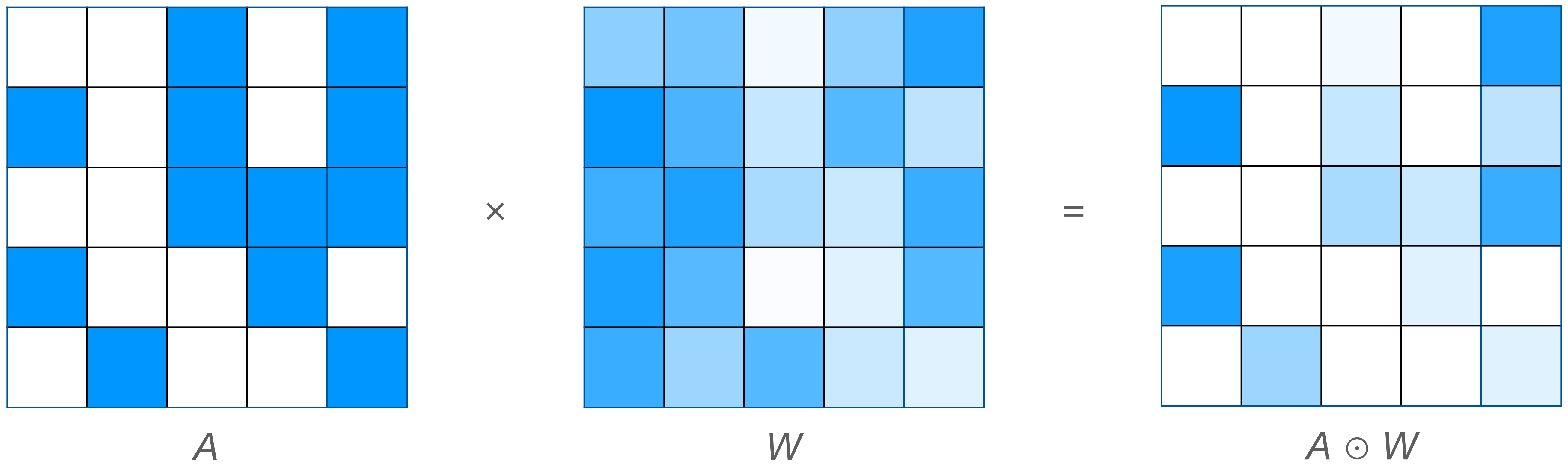
Weights & Adjacency Matrices

- “Spike-and-slab”

$$h_{n \to n'}(\Delta t) = a_{n \to n'} \cdot w_{n \to n'} \cdot \hbar(\Delta t; \theta_{n \to n'})$$

$$\boldsymbol{A} \in \{0, 1\}^{N \times N}$$

$$\boldsymbol{W} \in \mathbb{R}_{+}^{N \times N}$$

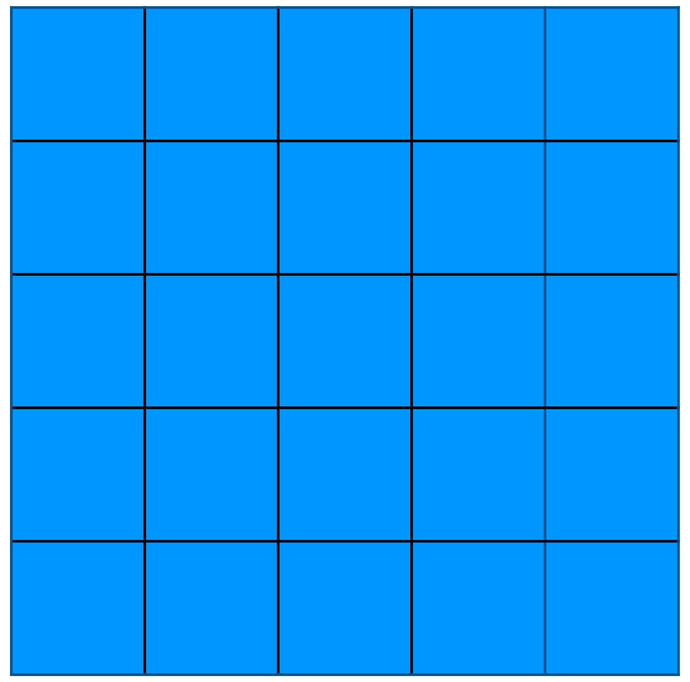


Hawkes Processes
Networks

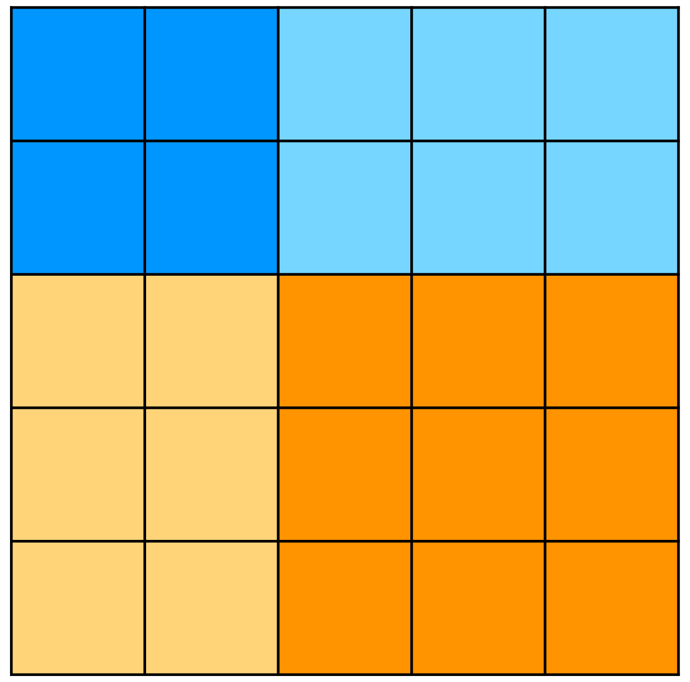
Name	$\boldsymbol{\vartheta}$	$\mathrm{dom}(\boldsymbol{z}_n)$	$\rho_{n\to n'}$
Empty Model	—	—	0
Dense Model	—	—	1
Bernoulli Model	ρ	—	ρ
Stochastic Block Model	$\{\{\rho_{k\to k'}\}\}$	$\{1,\ldots,K\}$	$\rho_{z_n\to z_{n'}}$
Latent Distance Model	γ_0	$\mathbb{R}^K$	$\sigma(-\lvert\lvert\boldsymbol{z}_n-\boldsymbol{z}_{n'}\rvert\rvert_2^2+\gamma_0)$

$$
\begin{aligned}
p(\boldsymbol{A}\mid\boldsymbol{z},\boldsymbol{\vartheta}) &= \prod_{n=1}^{N}\prod_{n'=1}^{N} p(a_{n\to n'}\mid \boldsymbol{z}_n,\boldsymbol{z}_{n'},\boldsymbol{\vartheta}) \\
&= \prod_{n=1}^{N}\prod_{n'=1}^{N} \mathrm{Bern}(a_{n\to n'}\mid \rho_{n\to n'}).
\end{aligned}
$$

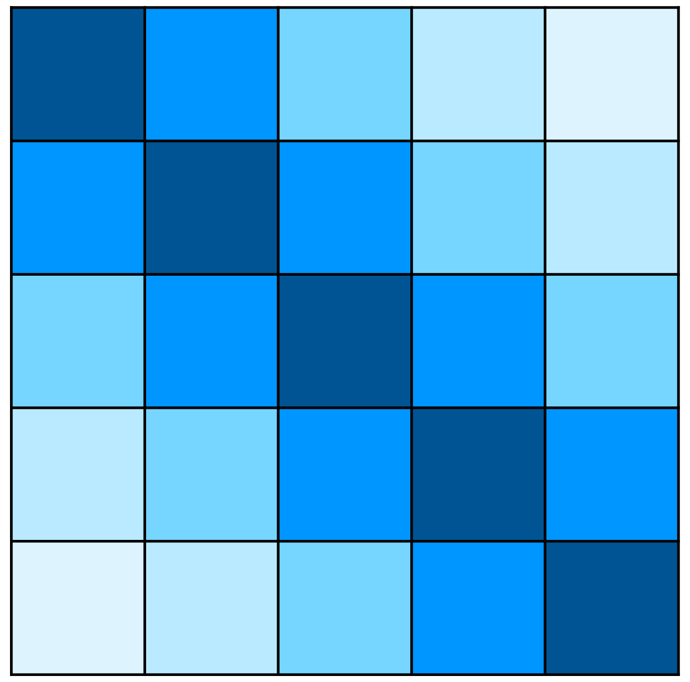
Bernoulli



Stochastic Block



Latent Distance



Package Features

Modular Design

- Processes are built from components: baseline processes, impulse responses, and network models.
- Components only need to adhere to a common interface, e.g., `rand, resample!`
- Abstract types provide common process methods, e.g., `rand, mle!, mcmc!`

```
mutable struct ContinuousStandardHawkesProcess
    baseline::Baseline
    impulses::ImpulseResponse
    weights::Weights
end
```

DATA-DRIVEN SOCIAL SCIENCE INITIATIVE

Package Features

Modular Design (Baselines)

- For example, any baseline process that implements the following interface is valid:

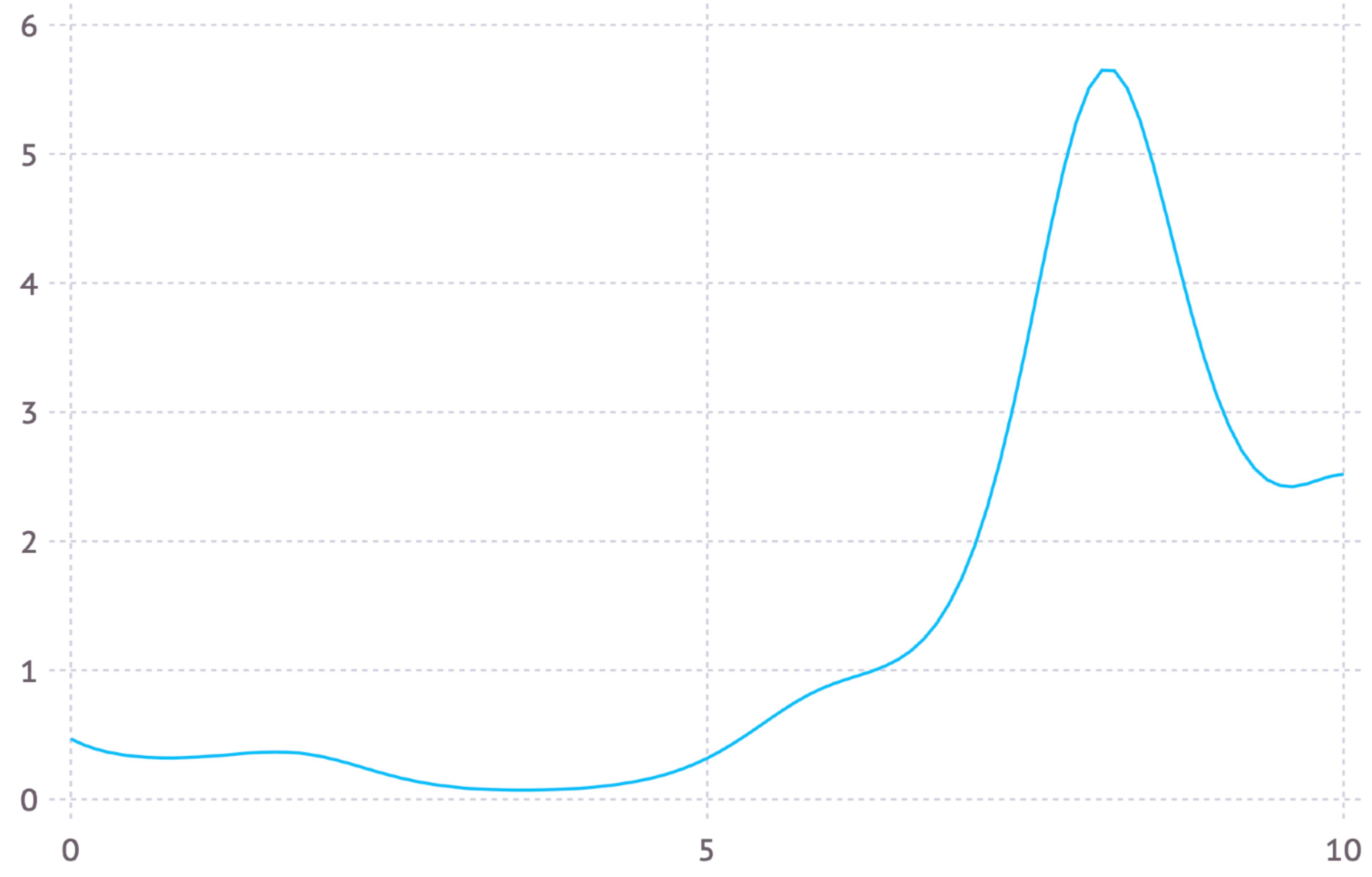
```
abstract type Baseline end

function ndims(baseline::Baseline) end
function params(baseline::Baseline) end
function params!(baseline::Baseline, ...) end
function rand(baseline::Baseline, ...) end
function resample!(baseline::Baseline, ...) end
function update!(baseline::Baseline, ...) end
function intensity(baseline::Baseline, ...) end
function integrated_intensity(baseline::Baseline, ...) end
function loglikelihood(baseline::Baseline, ...) end
function logprior(baseline::Baseline) end
```

DATA-DRIVEN SOCIAL SCIENCE INITIATIVE

Package Features

Modular Design (Baselines)

- The default baseline is a `HomogeneousProcess`
    - Simple conjugate model provides (fast) analytic Gibbs sampling
    - Constant intensity isn't appropriate for all settings (e.g., seasonalities)
- We provide an alternative non-homogeneous baseline, `LogGaussianCoxProcess`
    - Extremely flexible with included `Kernel` structs
    - Requires (slower) Metropolis-Hastings sampling



Package Features

Modular Design (Networks)

- Sparsity made possible by combining `ContinuousStandardHawkesProcess` with a `Network` model
- We provide this via `ContinuousNetworkHawkesProcess`
  - Add new parameters:
    - Adjacency matrix
    - Network
  - Lose maximum-likelihood estimation

```
mutable struct ContinuousNetworkHawkesProcess
    baseline::Baseline
    impulses::ImpulseResponse
    weights::Weights
    adjacency_matrix::Matrix
    network::Network
end
```

DATA-DRIVEN SOCIAL SCIENCE INITIATIVE

Package Features
Modular Design (Networks)

- Users can specify any network model that adheres to the `Network` interface
- Networks connect to the the overall model through adjacency matrix
- Currently available networks:
  - `DenseNetwork`
  - `BernoulliNetwork`
- Work in progress:
  - `StochasticBlockNetwork`
  - `LatentDistanceNetwork`

```
abstract type Network end

function size(network::Network) end
function params(network::Network) end
function rand(network::Network) end
function link_probability(network::Network) end
function resample!(network::Network, data) end
function update!(network::Network, data) end
function loglikelihood(network::Network, data) end
```

Package Features

Continuous and Discrete Processes

```
abstract type HawkesProcess end
```

```
abstract type ContinuousHawkesProcess end
```

```
abstract type DiscreteHawkesProcess end
```

events	nodes
0.1	1
0.3	1
0.9	2
1.2	1
1.3	2
2.1	2
2.5	1
4.1	1
4.3	2
4.8	1

event counts

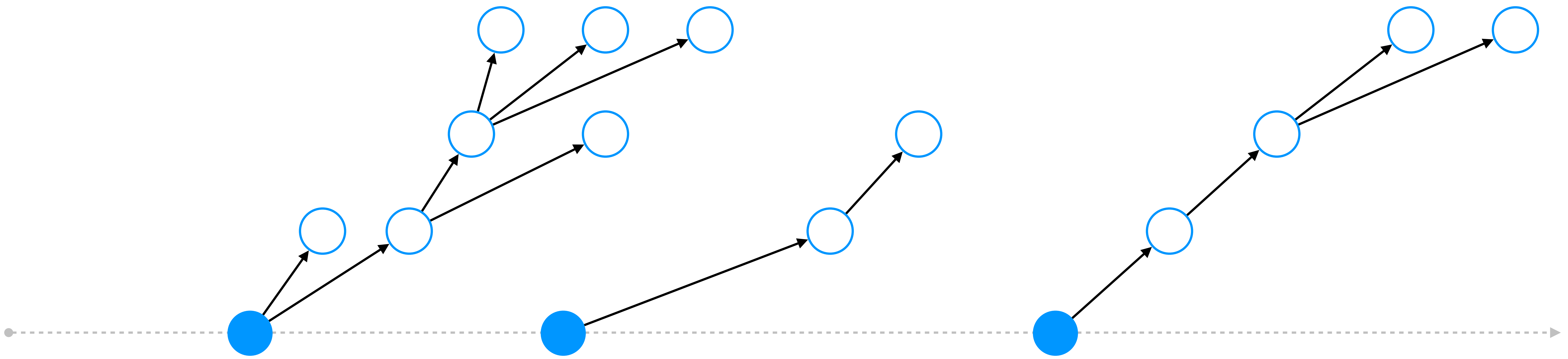
1		1		1	1		1		1
2	1	1	2	2	2	3	4	2	1
1	2	2	1	4	2	1	2	1	2
	3	2	2	6	4	3	2	2	1
1	2	1	3	2	2	2	1	3	6

Package Features

Simulation Method

- “Poisson thinning”

```
process = ContinuousStandardHawkesProcess(baseline, impulses, weights)
data = rand(process, duration)
```



Package Features

Estimation Methods

- Maximum-likelihood `mle!(process, data; regularized=false)`
  - Limited to standard models
- Markov chain Monte Carlo (Gibbs) `mcmc!(process, data; nsteps=1000)`
- Mean-field Variational Inference
  - Variational Bayes `vb!(process, data)`
  - Limited to discrete models

Package Features
Multithreading

- Resampling includes (embarrassingly) parallel calculations, e.g.
    - Block Gibbs sampling (adjacency matrix, auxiliary parents)
    - Metropolis-Hasting sampling (baseline intensity)
- Loglikelihood calculations include (embarrassingly) parallel calculations (total intensity)
- Julia `Base` includes submodule `Threads` provides `@threads` macro for parallel for-loops:

```
Threads.@threads for index in eachindex(events)
    # regular for-loop work
end
```

DATA-DRIVEN SOCIAL SCIENCE INITIATIVE

Package Status

- Work in Progress
  - Univariate and independent processes
- Feature Requests
  - Advanced network models (e.g., stochastic block, latent distance, time-varying)
  - Stochastic variational inference
  - Exogenous covariates
  - Utility methods (e.g., model initialization, evaluation, and visualization, etc.)

Thank you!

https://cswaney.github.io/NetworkHawkesProcesses.jl/dev

DATA-DRIVEN SOCIAL SCIENCE INITIATIVE

https://ddss.princeton.edu