SAND2022-12316C

Sandia National Laboratories

Exceptional service in the national interest

WAKE FOREST UNIVERSITY

MathSci.ai

Generalized CP (GCP) Tensor Decompositions

Grey Ballard, ***Daniel M. Dunlavy***, Tamara G. Kolda

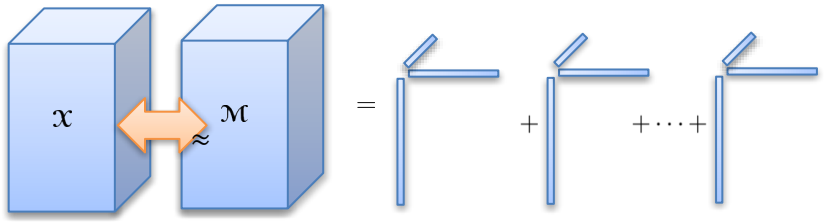
SIAM MDS22, Tensor Minitutorial

September 29, 2022

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Revisiting CP: Generalizing the Goodness-of-Fit Criteria



$$\mathcal{X} \approx \mathcal{M} = \sum_{j=1}^{r} \mathbf{a}_j \circ \mathbf{b}_j \circ \mathbf{c}_j = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \sum_i (x_i - m_i)^2 \text{ s.t. } \mathcal{M} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

$$F(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \sum_i (x_i - m_i)^2$$

$$F(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \sum_i f(x_i, m_i)$$

generalized loss function

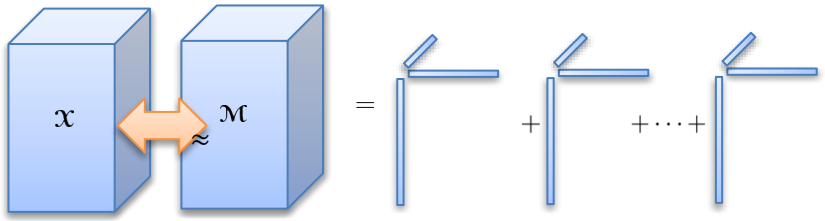
GCP: Generalized CP Tensor Decompositions

Hong, Kolda & Duersch (2020)

Matlab Tensor Toolbox:

`gcp_opt`

Poisson CP: Identity Link

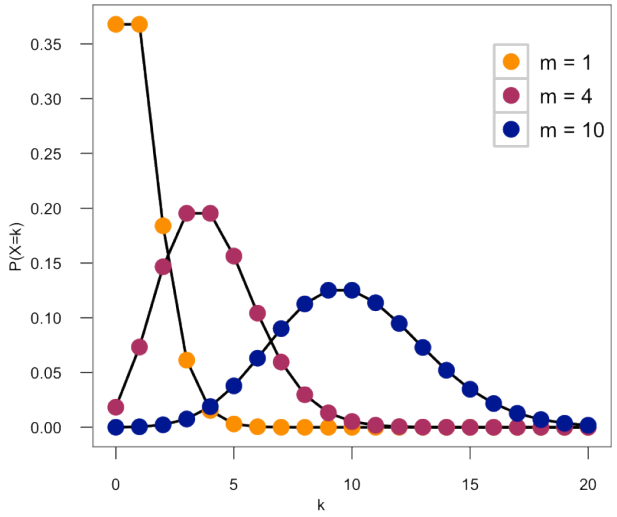


Consider data to be Poisson distributed with parameter m_i

$$x_i \sim \text{Poisson}(m_i)$$

Probability Mass Function (PMF):

$$\frac{e^{-\lambda}\lambda^x}{x!}$$



Minimizing negative log likelihood with $\lambda = m$:

$$\min F(\mathcal{X}, \mathcal{M}) = \sum_i m_i - x_i \log(m_i) + \log(x_i!)$$

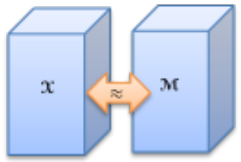
Requires $m_i > 0$

$$\mathbb{E}(X_i) = m_i$$

CP-APR: CP-Alternating Poisson Regression Tensor Decompositions

Chi & Kolda (2012); Hansen, Plantenga & Kolda (2015)

GCP: Loss Functions



$$\min \quad F(\mathcal{X}, \mathcal{M}) = \sum_i f(x_i, m_i)$$
$$\text{s.t.} \quad \mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_d]\!]$$

Loss Function Name	Data	`'type'` Parameter in `gcp_opt`
Gaussian	real-valued	`'normal', 'gaussian'`
Poisson with Linear Link	count	`'count', 'poisson'`
Poisson with Log Link	count	`'poisson-log'`
Bernoulli with Odds Link	binary	`'binary', 'bernoulli-odds'`
Bernoulli with Logit Link	binary	`'bernoulli-logit'`
Rayleigh	real-valued	`'rayleigh'`
Gamma	nonnegative real-valued	`'gamma'`
Huber	nonnegative real-valued	`'huber (<d>)'`
Negative Binomial	count	`'negative-binomial (<r>)'`
Beta	nonnegative real-valued	`'beta (<b>)'`

Advanced: Create and use your own loss function (and gradient)

Hong, Kolda & Duersch (2020)

GCP: Loss Function and Gradient Computations

Given data tensor $\mathcal{X}$ and current model $\mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_d]\!]$.

1: $F \leftarrow \sum_i f(x_i, m_i)$
2: Compute $\mathcal{G}$ based on current $\mathcal{M}$, i.e., $g_i = \frac{\partial f}{\partial m}(x_i, m_i)$ for all i
3: **for** $k = 1, \ldots, d$ **do**
4: $\quad \Delta_k \leftarrow \mathbf{G}_{(k)}(\mathbf{A}_d \odot \cdots \odot \mathbf{A}_{k+1} \odot \mathbf{A}_{k-1} \odot \cdots \odot \mathbf{A}_1)$
5: **end for**

MTTKRP

Compute Gradient

Important: Computing the GCP gradient is expensive!

Hong, Kolda & Duersch (2020)

GCP: Optimization Methods

Optimization Method	Notes	'opt' in gcp_opt
Limited-Memory Quasi-Newton with Bound Constraints	Default for dense tensor data	'lbfgsb'
Stochastic gradient descent (SGD)		'sgd'
ADAM: adaptive moment estimation	Default for sparse tensor data	'adam'
AdaGrad: adaptive gradient		'adagrad'

Stochastic Methods: User must specify amount of sampling for good performance

Sampling Method	Notes	'sampler' in gcp_opt
Uniform	Entries are selected uniformly at random. Default for dense tensors.	'uniform'
Stratified	Zeros and nonzeros sampled separately. Default for sparse tensors.	'stratified'
Semi-Stratified	Modification to stratified sampling that avoids rejection sampling for better efficiency at the cost of potentially higher variance.	'semi-stratified'

Advanced: Create and use your own optimizer

Hong, Kolda & Duersch (2020); Kolda & Hong (2020)

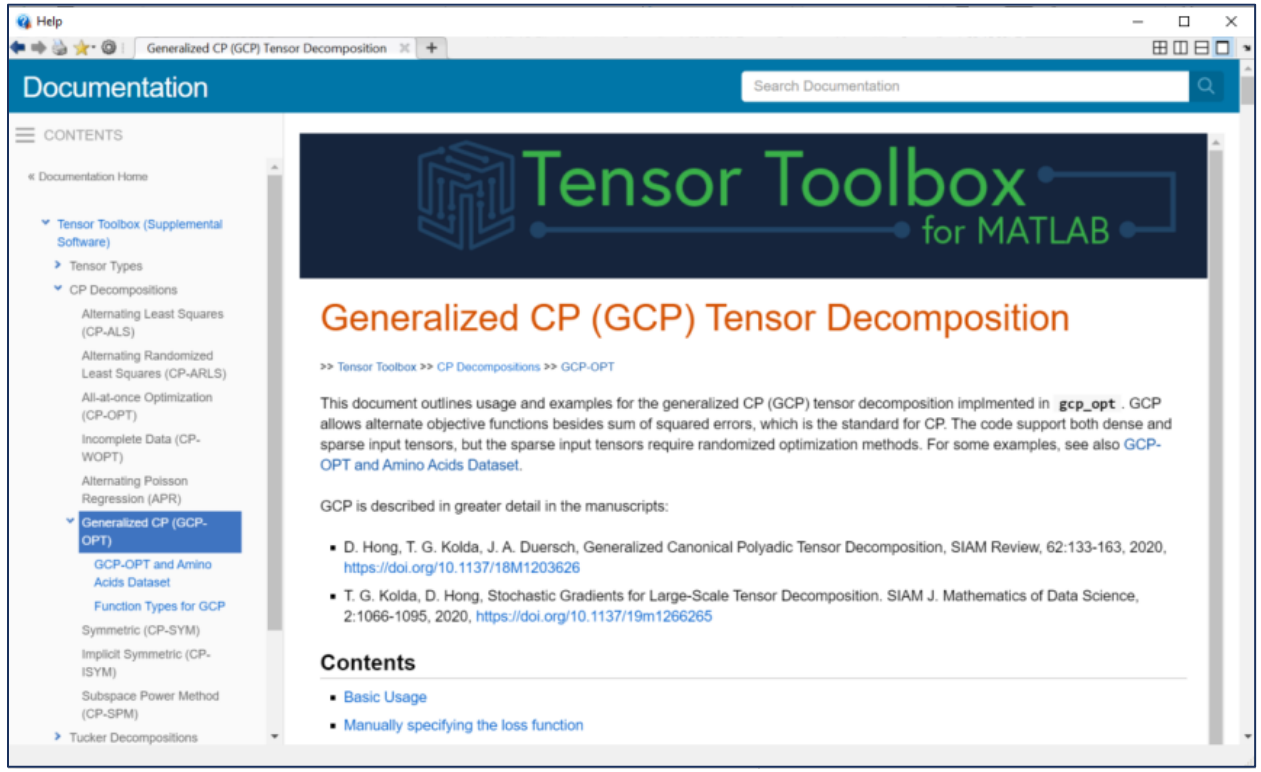
GCP: Matlab Tensor Toolbox

GCP Parameters: Many user parameters that impact time to solution and accuracy

```
>> help gcp_opt
```

```
gcp_opt Fits Generalized CP decomposition with user-specified function.

  M = gcp_opt(X,R,'type',TYPE) computes an estimate of the best rank-R
  generalized CP (GCP) decomposition of the tensor X for the specified
  generalized loss function specified by TYPE. The input X can be a
  tensor or sptensor. The result M is a ktensor. A full set of types can
  be found in GCP_FG_SETUP, but popular options include

     'binary' - Bernoulli distribution for binary data
     'count'  - Poisson distribution for count data (see also CP_APR)
     'normal' - Gaussian distribution (see also CP_ALS and CP_OPT)
     'huber (0.25)' - Similar to Gaussian but robust to outliers
     'rayleigh' - Rayleigh distribution for nonnegative data
     'gamma'  - Gamma distribution for nonnegative data

  M = gcp_opt(X,R,'func',FH,'grad',GH,'lower',LB) passes a user-specified
  choice for the elementwise loss function, corresponding gradient, and
  lower bound on the factor matrix entries. This is an alternative to
  specifying the 'type' as shown above.

  M = gcp_opt(X,R,...,'opt',OPT) specifies the optimization method:
     'lbfgsb' - Bound-constrained limited-memory BFGS
     'sgd' - Stochastic gradient descent (SGD)
     'adam' - Momentum-based SGD method
  If X is dense, any of the three options can be used, and 'lbfgsb' is
  the default. The X is sparse, only 'sgd' and 'adam' (default) are
  options. Each method has specific parameters, see the documentation for
```

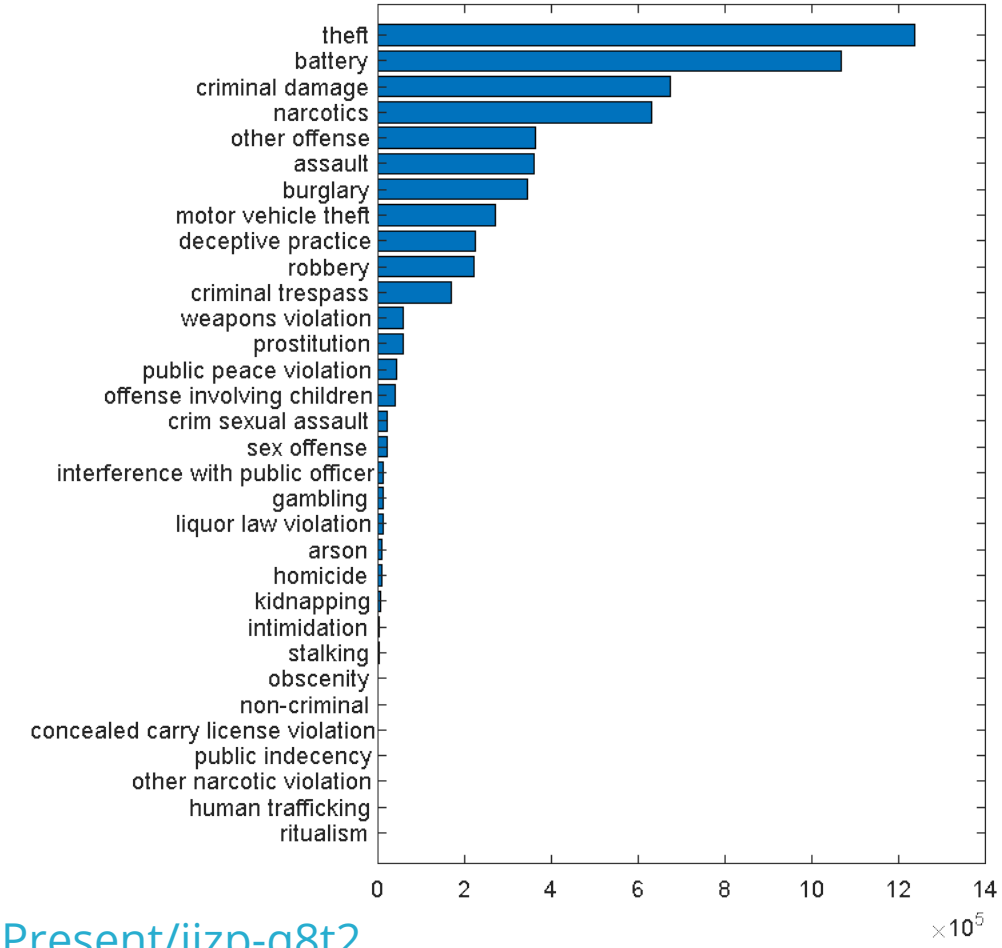
```
>> doc tensor_toolbox
```



GCP Example: Chicago Crime Data

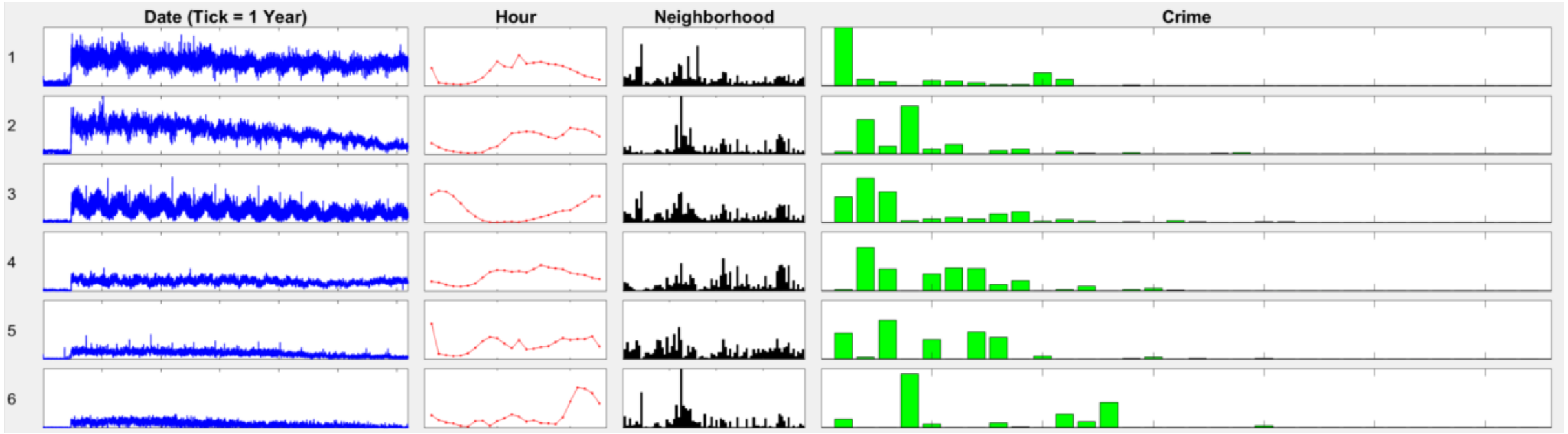
- 4-way count tensor
  - 6,186 Days
  - 24 Hours of the Day
  - 77 Community Areas
  - 32 Crime Types
- Non-zeros: 5,330,673
  - 0.21GB for sparse tensor
- Distribution of entries
  - 0: 98.54%
  - 1: 1.33%
  - > 2: 0.12%
- Obtained from FROSTT
  - http://frosttio/tensors/chicago-crime/
- Data originally from Chicago Data Portal
  - https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-Present/ijzp-q8t2

GCP Example
optimizer = ADAM
loss function type = count
$f(x, m) = m - x \log m$
rank = 10
sampling = stratified
function samples = 10,000
gradient samples = 3,500



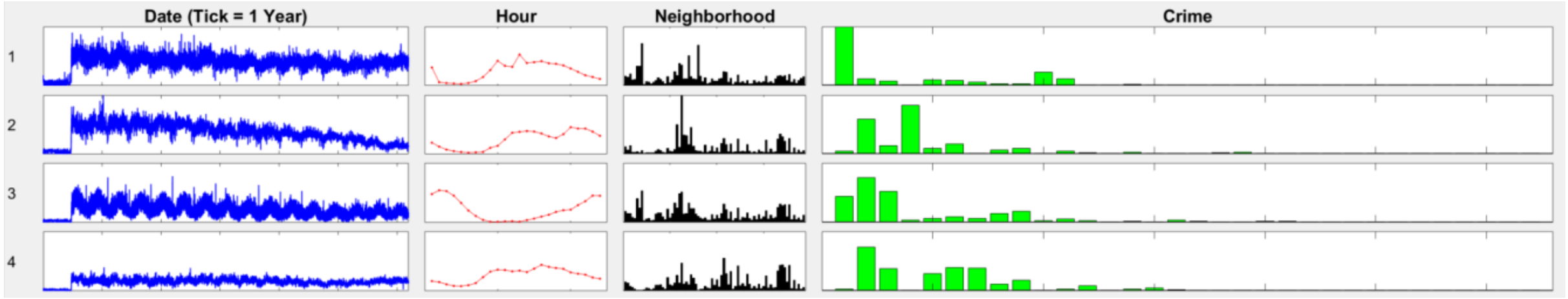
GCP Example: Chicago Crime Data

```
>> load chicago_crime.mat                % load data from file
>> X = crime_tensor;                     % extract sparse tensor from data
>> M = gcp_opt(X,10,'type','count');     % compute GCP decomposition
>> chicago_viz(M,1);                     % visualize results
```

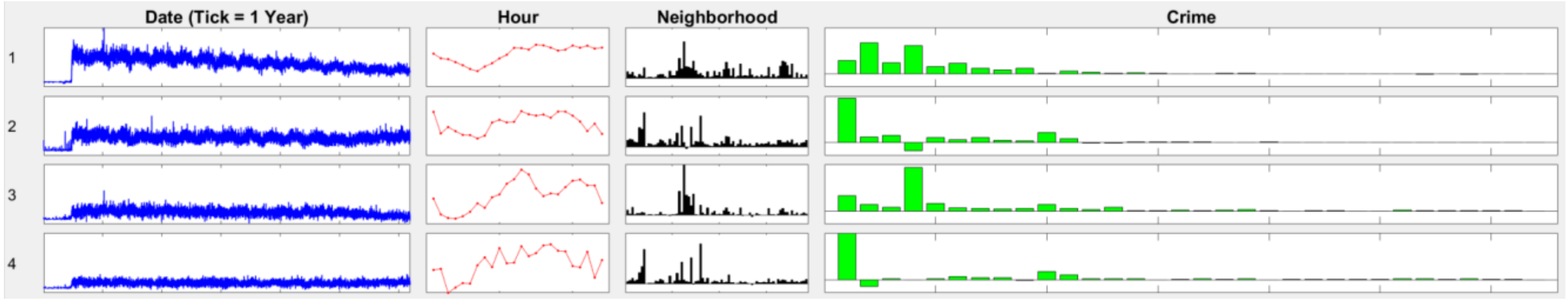


GCP Example: Chicago Crime Data

```
>> M = gcp_opt(X,10,'type','count’);
```

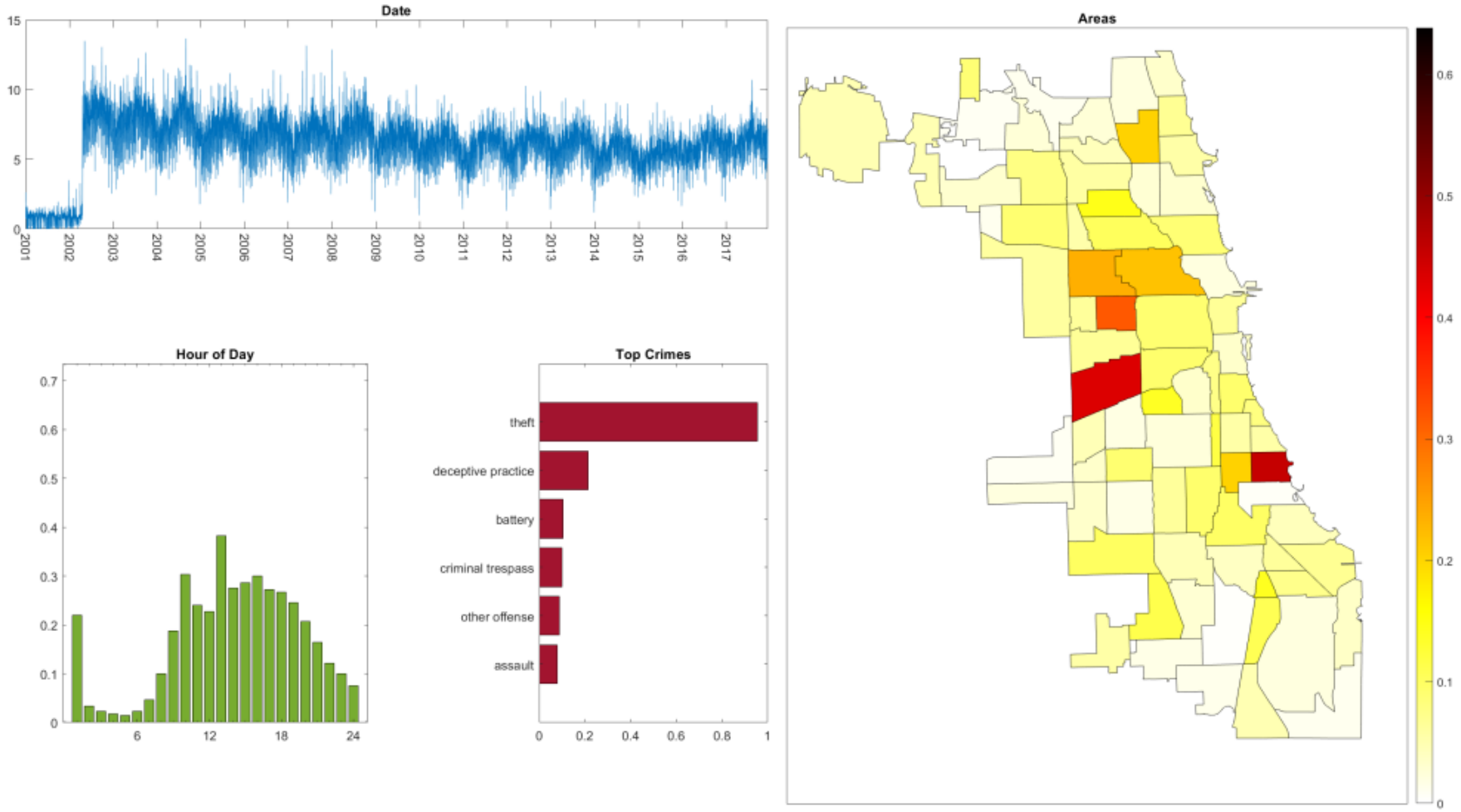



```
>> M = gcp_opt(X,10,'type’,’normal’);
```



GCP Example: Chicago Crime Data (Component #1)

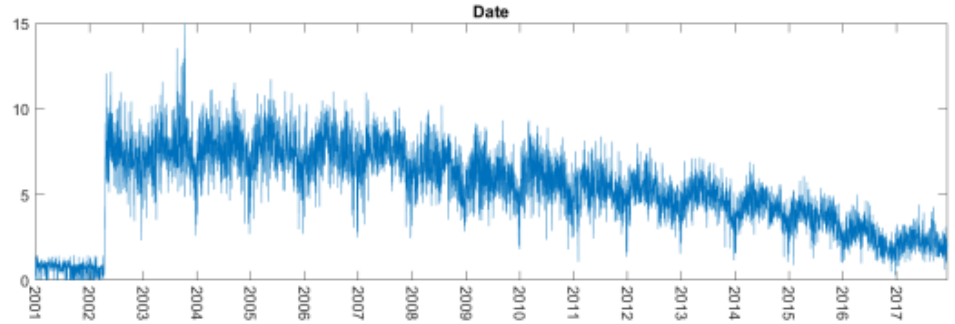
```
>> chicago_component_viz(M,1,1);
```

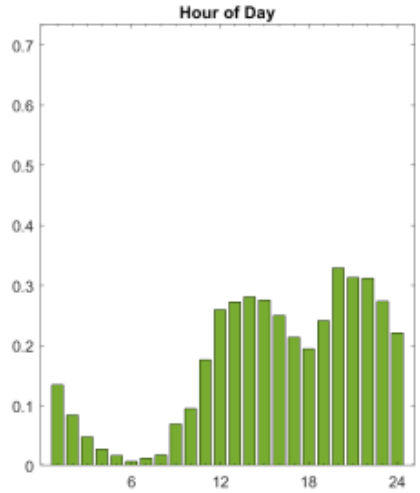


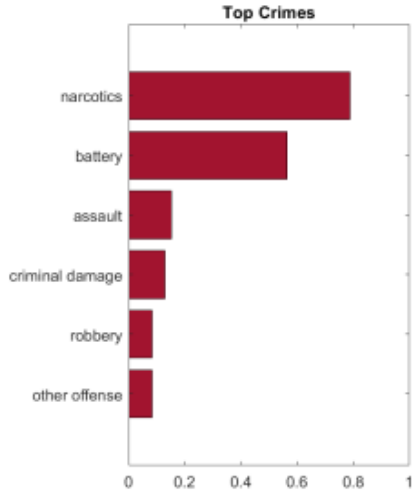
Shape file for areas downloaded from https://data.cityofchicago.org/Facilities-Geographic-Boundaries/Boundaries-Community-Areas-current-/cauq-8yn6

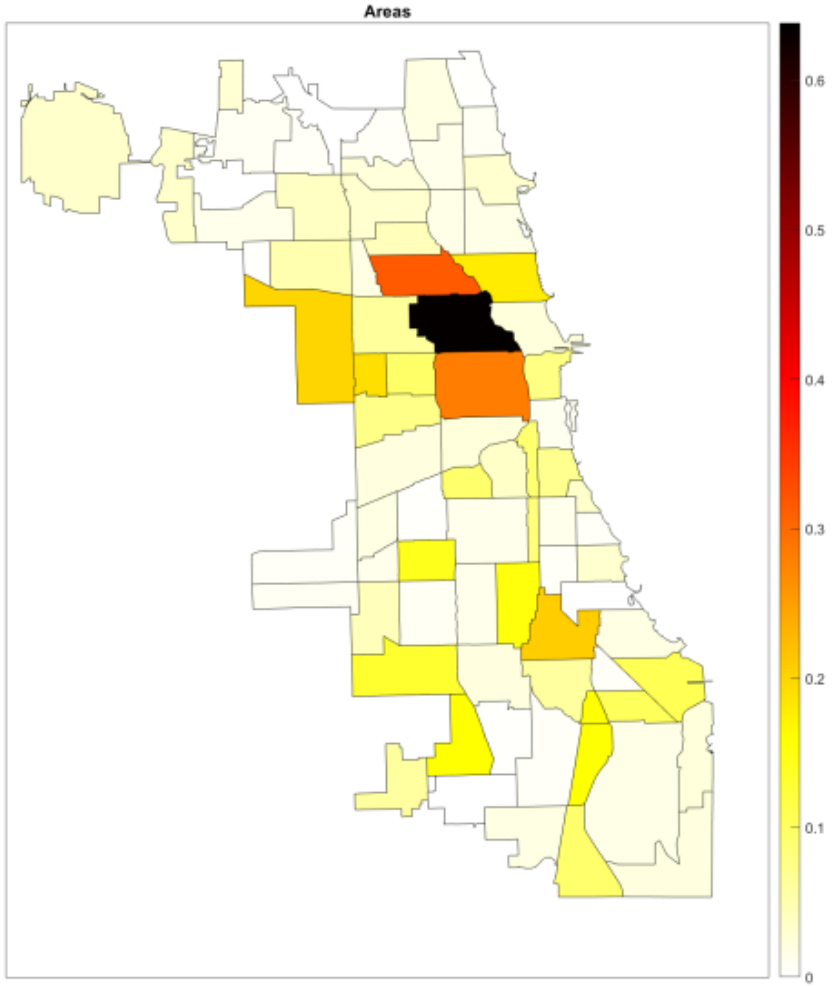
GCP Example: Chicago Crime Data (Component #2)

```
>> chicago_component_viz(M,2,1);
```









Shape file for areas downloaded from https://data.cityofchicago.org/Facilities-Geographic-Boundaries/Boundaries-Community-Areas-current-/cauq-8yn6

References

- Chi & Kolda (2012). On Tensors, Sparsity, and Nonnegative Factorizations, *SIAM Journal on Matrix Analysis and Applications*, 33 (4), 1272-1299.
- Hansen, Plantenga & Kolda (2015). Newton-Based Optimization for Kullback-Leibler Nonnegative Tensor Factorizations, *Optimization Methods and Software*, 30 (5), 1002-1029.
- Hong, Kolda & Duersch (2020). Generalized Canonical Polyadic Tensor Decomposition. *SIAM Review*, 62 (1), 133-163.
- Kolda & Hong (2020). Stochastic Gradients for Large-Scale Tensor Decomposition. *SIAM Journal on Mathematics of Data Science*, 2 (4), 1066-1095.