

Using low-rank tensor formats to enable computations of cancer progression models in large state spaces

Simon Pfahler¹, Peter Georg¹, Y. Linda Hu², Stefan Vocht², Rudolf Schill^{2,3}, Andreas Lösch², Kevin Rupp^{2,3}, Stefan Hansch², Maren Klever⁴, Lars Grasedyck⁴, Rainer Spang², Tilo Wettig¹



Scan for digital version

¹Department of Physics, University of Regensburg

²Department of Informatics and Data Science, University of Regensburg

³Department of Biosystems Science and Engineering, ETH Zürich

⁴Institute for Geometry and Applied Mathematics, RWTH Aachen University

Summary

- Comprehensive cancer progression models should include a large number d of genomic events
- Mutual Hazard Networks model the progression process using only d^2 parameters [1]
 - Computational complexity of a straightforward implementation still scales exponentially in d
 - Calculations using $\gtrsim 25$ events are computationally infeasible [2]
- Tensor Trains allow for cost-effective storage and calculations for high-dimensional tensors
 - This method reduces the computational complexity from exponential to polynomial in d

Mutual Hazard Network (MHN) model

- MHN models tumor progression as a continuous-time Markov chain on the 2^d -dimensional state space of possibly active events [1]
- Events can only occur one at a time
- Transition rates are given by

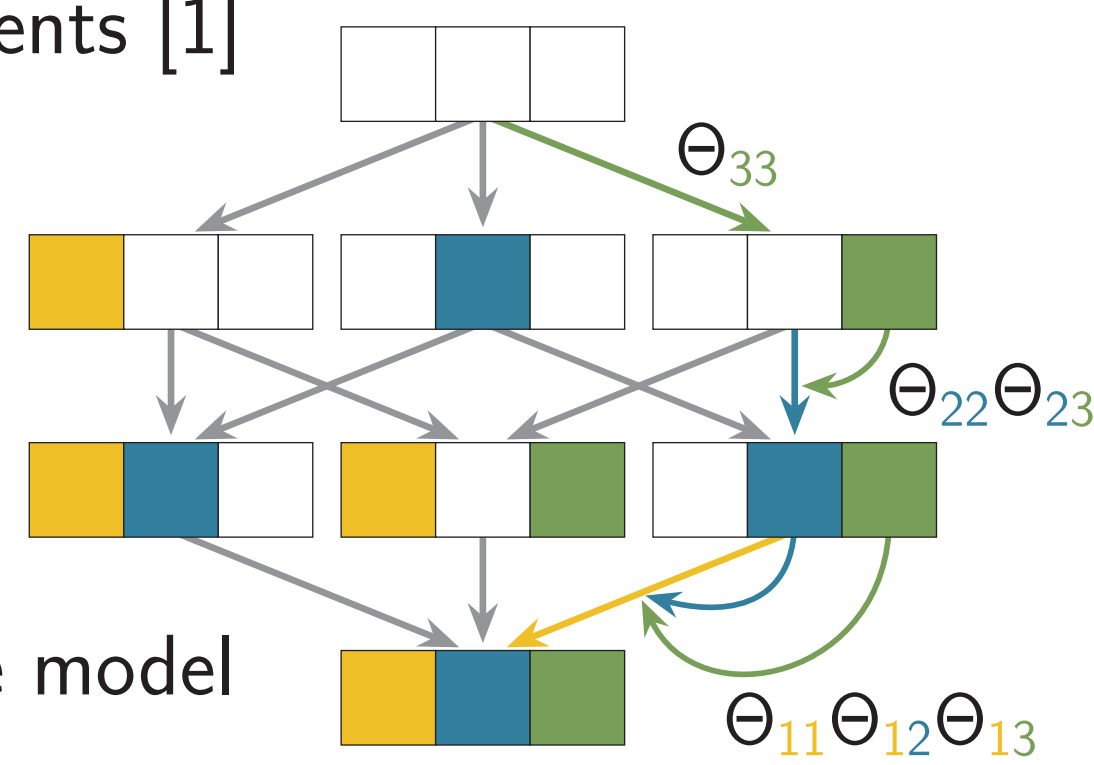
$$(\mathbf{Q}_\Theta)_{\mathbf{x} \rightarrow \mathbf{x}+i} = \underbrace{\Theta_{ij}}_{\text{base rate}} \prod_{j=1}^d \underbrace{\Theta_{ij}}_{\text{influences}}$$
- $\mathbf{Q}_\Theta \in \mathbb{R}^{2^d \times 2^d}$ is sparse, $\Theta \in \mathbb{R}^{d \times d}$ describes the model in compact form using only d^2 parameters
- Time marginal probability distribution from Θ :

$$\mathbf{p}_\Theta = (\mathbf{Id} - \mathbf{Q}_\Theta)^{-1} \mathbf{p}_\emptyset \quad \mathbf{p}_\emptyset = (100\%, 0\%, 0\%, \dots, 0\%)^T$$
- Optimal Θ matrices are found by optimizing the time-marginalized Kullback-Leibler divergence from the given data distribution $\mathbf{p}_\mathcal{D}$:

$$S_{\text{KL}}(\mathbf{p}_\Theta) = \sum_{\mathbf{x}} (\mathbf{p}_\mathcal{D})_{\mathbf{x}} \log((\mathbf{p}_\Theta)_{\mathbf{x}})$$
- Gradients can be calculated analytically:

$$\frac{\partial S_{\text{KL}}}{\partial \Theta_{ij}} = \sum_{\mathbf{y}, \mathbf{z}} \underbrace{\frac{\partial S_{\text{KL}}}{\partial (\mathbf{p}_\Theta)_{\mathbf{x}}} (\mathbf{Id} - \mathbf{Q}_\Theta)^{-1}}_{=\mathbf{q}_{\mathbf{y}}} \left(\frac{\partial \mathbf{Q}_\Theta}{\partial \Theta_{ij}} \right)_{\mathbf{yz}} (\mathbf{p}_\Theta)_{\mathbf{z}}$$
- KL divergence and gradient calculation time is dominated by solution time of two linear equations:

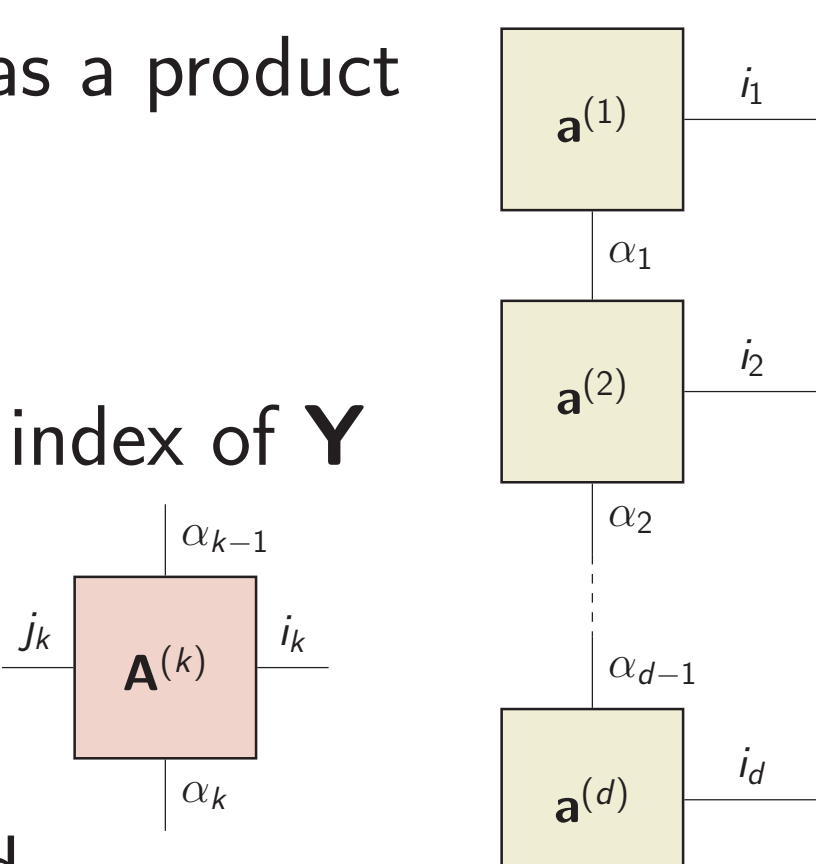
$$(\mathbf{1} - \mathbf{Q}_\Theta) \mathbf{p}_\Theta = \mathbf{p}_\emptyset \quad (\mathbf{1} - \mathbf{Q}_\Theta)^T \mathbf{q} = \frac{\partial S_{\text{KL}}}{\partial \mathbf{p}_\Theta}$$



Tensor Train (TT) representation

- A d -dimensional tensors $\mathbf{a} \in \mathbb{C}^{n_1 \times \dots \times n_d}$ can be written as a product of d Tensor Train cores $\mathbf{a}^{(k)} \in \mathbb{C}^{r_{k-1} \times n_k \times r_k}$:

$$\mathbf{a}(i_1, \dots, i_d) = \mathbf{a}^{(1)}(i_1) \circ \dots \circ \mathbf{a}^{(d)}(i_d)$$
- $\mathbf{X} \circ \mathbf{Y}$ denotes contraction of last index of \mathbf{X} with first index of \mathbf{Y}
- Similar for operators $\mathbf{A} \in \mathbb{C}^{(m_1 \times \dots \times m_d) \times (n_1 \times \dots \times n_d)}$: TT cores $\mathbf{A}^{(k)} \in \mathbb{C}^{r_{k-1} \times m_k \times n_k \times r_k}$
- Storage cost is reduced from exponential to linear in d , but additional dependency on TT ranks r_k is introduced
- Many arithmetic operations can be performed directly in the TT format, reducing the computational complexity [3]:
 - Superposition $\lambda \mathbf{a} + \nu \mathbf{b}$: $\mathcal{O}(dn(r_{\mathbf{a}} + r_{\mathbf{b}})^2)$
 - Inner product $\langle \mathbf{a}, \mathbf{b} \rangle$: $\mathcal{O}(dn r_{\mathbf{a}} r_{\mathbf{b}} (r_{\mathbf{a}} + r_{\mathbf{b}}))$
 - Operator-by-Tensor product $\mathbf{A} \mathbf{b}$: $\mathcal{O}(dmn(r_{\mathbf{A}} r_{\mathbf{b}})^2)$
- Linear equations $\mathbf{A} \mathbf{x} = \mathbf{b}$ can also be solved efficiently directly in this format



References

- [1] R. Schill, S. Solbrig, T. Wettig, and R. Spang, *Modelling cancer progression using Mutual Hazard Networks*, *Bioinformatics* **36** (January, 2020) 241.
- [2] P. Georg, L. Grasedyck, M. Klever, R. Schill, R. Spang, and T. Wettig, *Low-rank tensor methods for Markov chains with applications to tumor progression models*, *Journal of Mathematical Biology* **86** (December, 2022).
- [3] P. Georg, *Tensor Train Decomposition for solving high-dimensional Mutual Hazard Networks*, PhD thesis, Universität Regensburg, October, 2022.

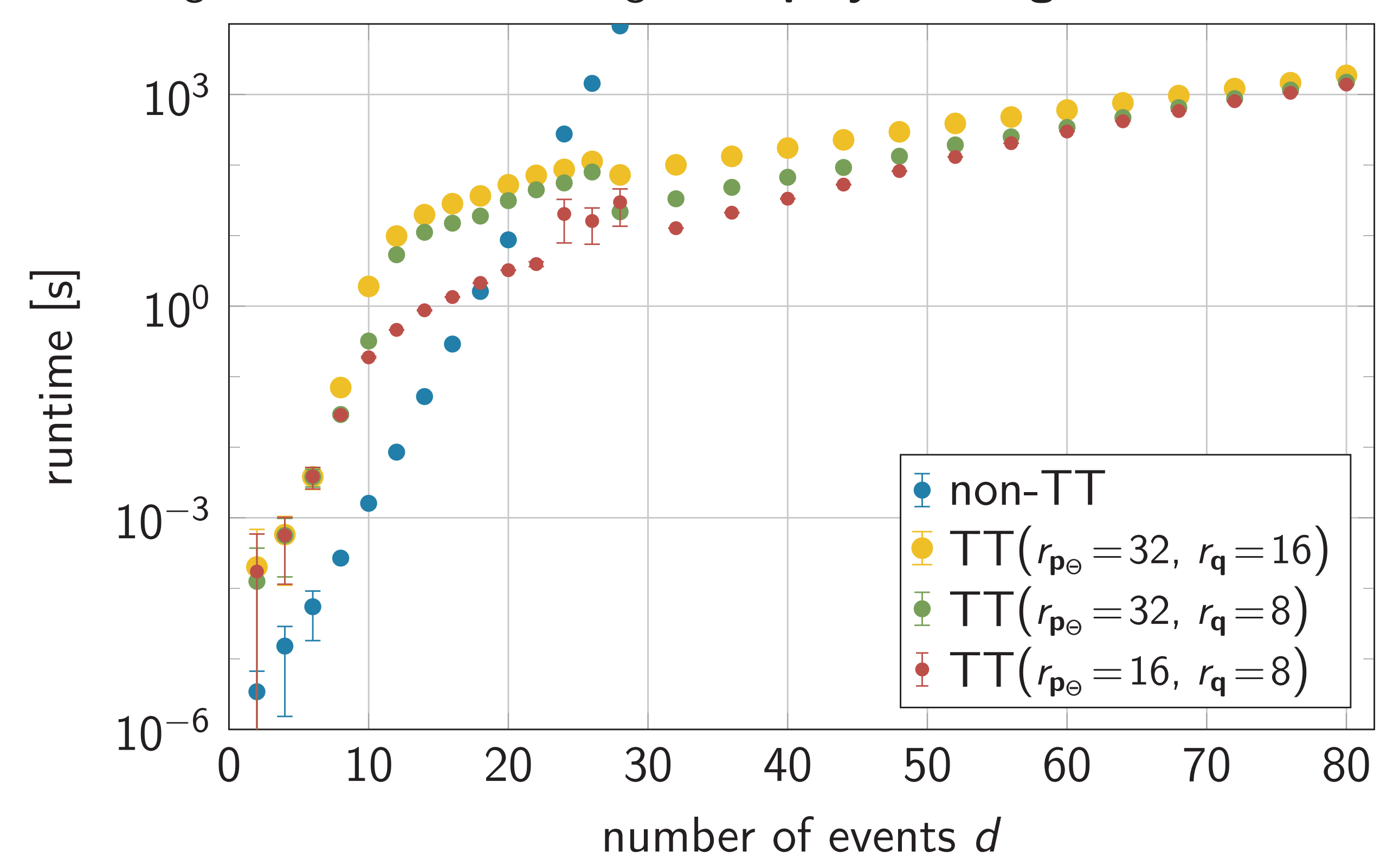
Tensor Trains for MHN

- Events are binary $\rightarrow n_k = 2$ for all mode sizes
- \mathbf{Q}_Θ can naturally be written as a Tensor Train [1]:

$$\mathbf{Q}_\Theta = \sum_{i=1}^d \left(\bigotimes_{j=1}^{i-1} \begin{pmatrix} 1 & 0 \\ 0 & \Theta_{ij} \end{pmatrix} \right) \otimes \begin{pmatrix} -\Theta_{ii} & 0 \\ \Theta_{ii} & 0 \end{pmatrix} \otimes \left(\bigotimes_{j=i+1}^d \begin{pmatrix} 1 & 0 \\ 0 & \Theta_{ij} \end{pmatrix} \right)$$
 - \mathbf{Q}_Θ is a sum of d rank-1 Tensor Trains
 - All TT-ranks of \mathbf{Q}_Θ are equal to d
- \mathbf{p}_\emptyset is a canonical unit Tensor Train, with all TT-ranks equal to 1
- \mathbf{p}_Θ and \mathbf{q} can be calculated in the TT format (max. TT ranks $r_{\mathbf{p}_\Theta}$ and $r_{\mathbf{q}}$)
- For gradients, each nonzero entry in $\mathbf{p}_\mathcal{D}$ has to be treated individually
 - One linear equation has to be solved for each nonzero entry (usually ~ 1000)
 - This can be parallelized trivially

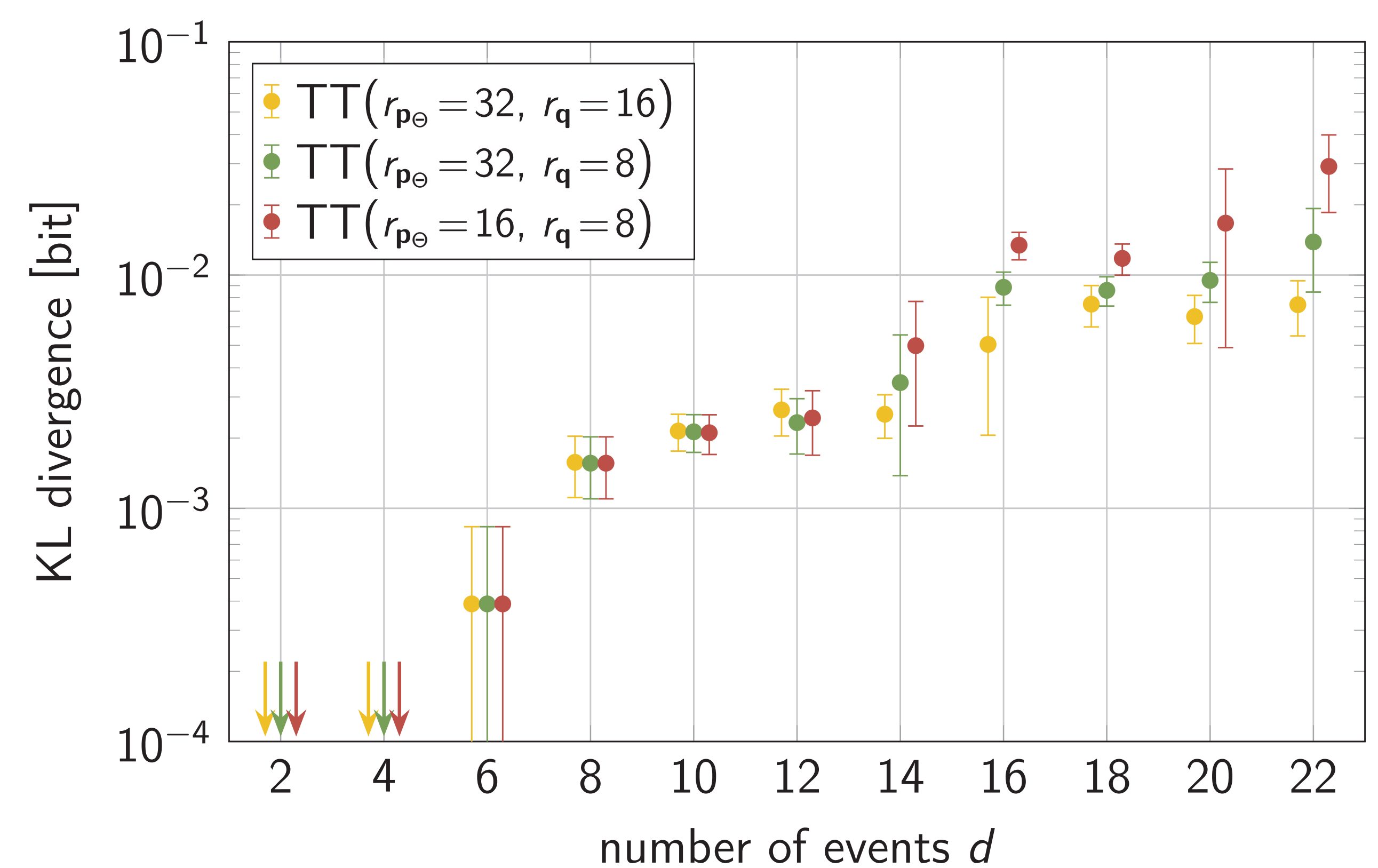
Results: Runtime speedup

- Runtime for one score and gradient evaluation
- $\mathbf{p}_\mathcal{D}$ constructed from 1000 random samples, Θ = independence model
- Runtime grows with $\sim d^{5.4}$ for large $d \Rightarrow$ **polynomial growth!**



Results: Accuracy of the TT solution

- KL divergence from exact result to TT solution after full optimization of Θ



Code availability

- C++ library for TT-calculations pRC: gitlab.com/pjgeorg/pRC
- Application-specific C++ library cMHN that utilizes pRC for MHN-calculations: soon to be open-source

Future improvements

- Reduce runtime by accelerating solution of linear equations in TT format
- Include formation of metastases in the model