Leveraging Recursive Gumbel-Max Trick for Approximate Inference in Combinatorial Spaces

Kirill Struminsky[1*], Artyom Gadetsky[1*], Denis Rakitin[1,4*], Danil Karpushkin[2,3,5*], Dmitry Vetrov[1,3]

Structured Latent Variable Models

Latent variable models can incorporate prior knowledge about data. Structured variables include **matching, graphs, subsets**, etc.

Recent structured variables extend the Gumbel-Softmax trick

- Define perturbed sampling algorithm
- Optimise a differentiable approximation of the sampler

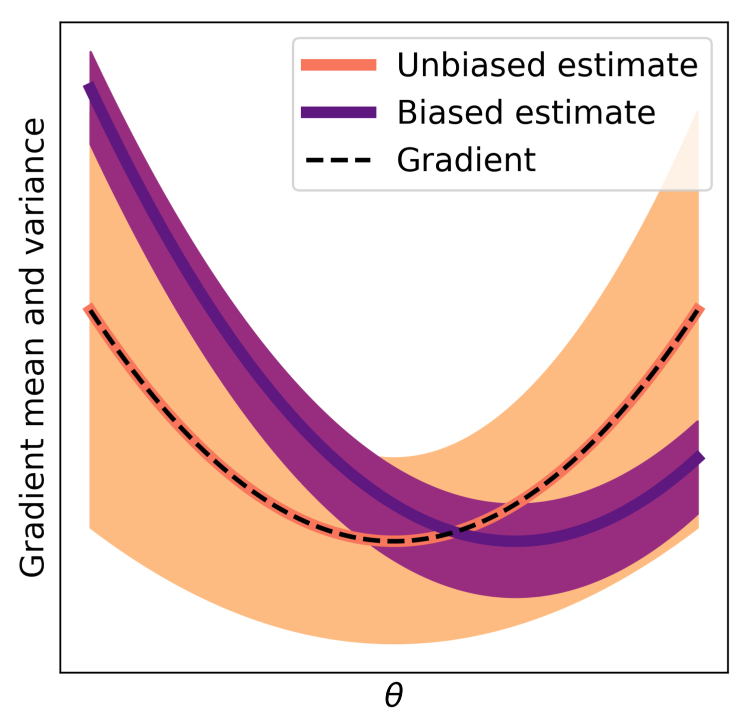
Instead, we design a template for sampling algorithms and optimise with score function estimators. We apply the template to define distributions over **permutations, matchings, binary trees, spanning trees,** etc.

Gradient Estimators

Tradeoff Between the Bias and the Variance of the Estimate

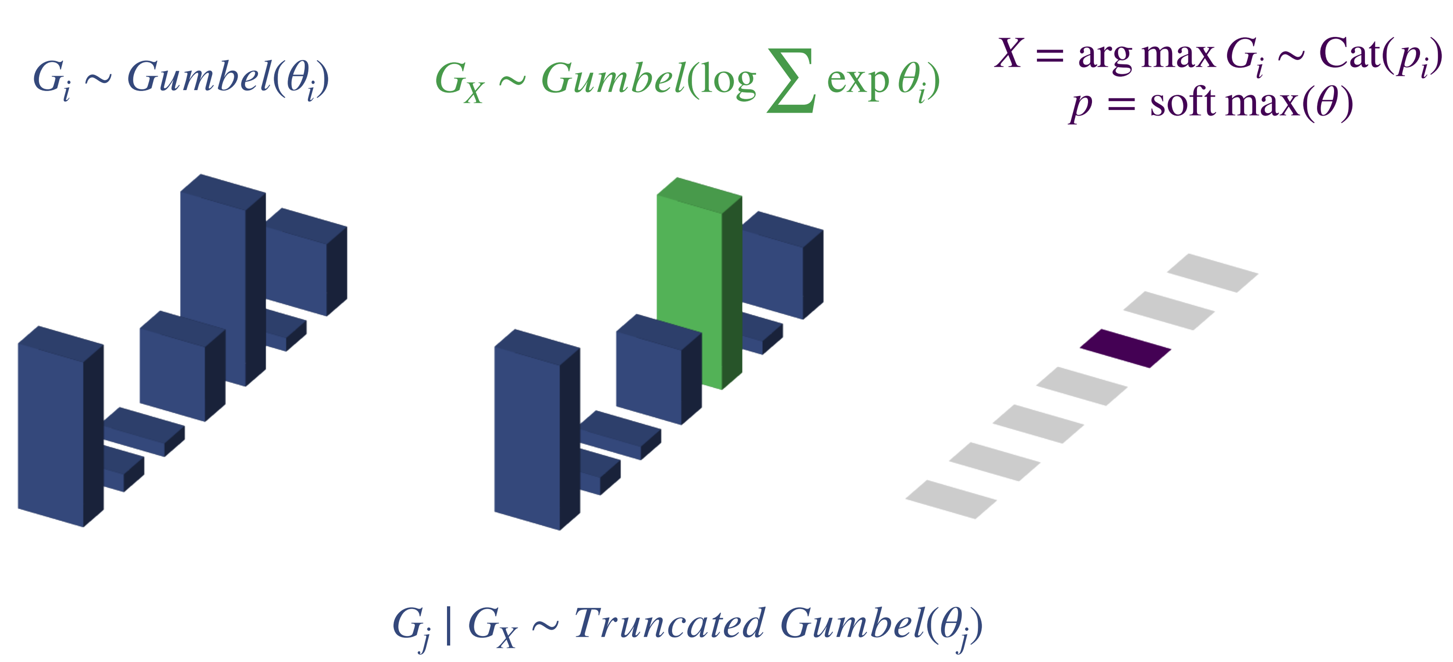
Relaxation-based estimates modify the original objective to reduce variance and to accelerate inference. The new objective leads estimation bias.

Score function estimates do not have such bias but have higher variance.



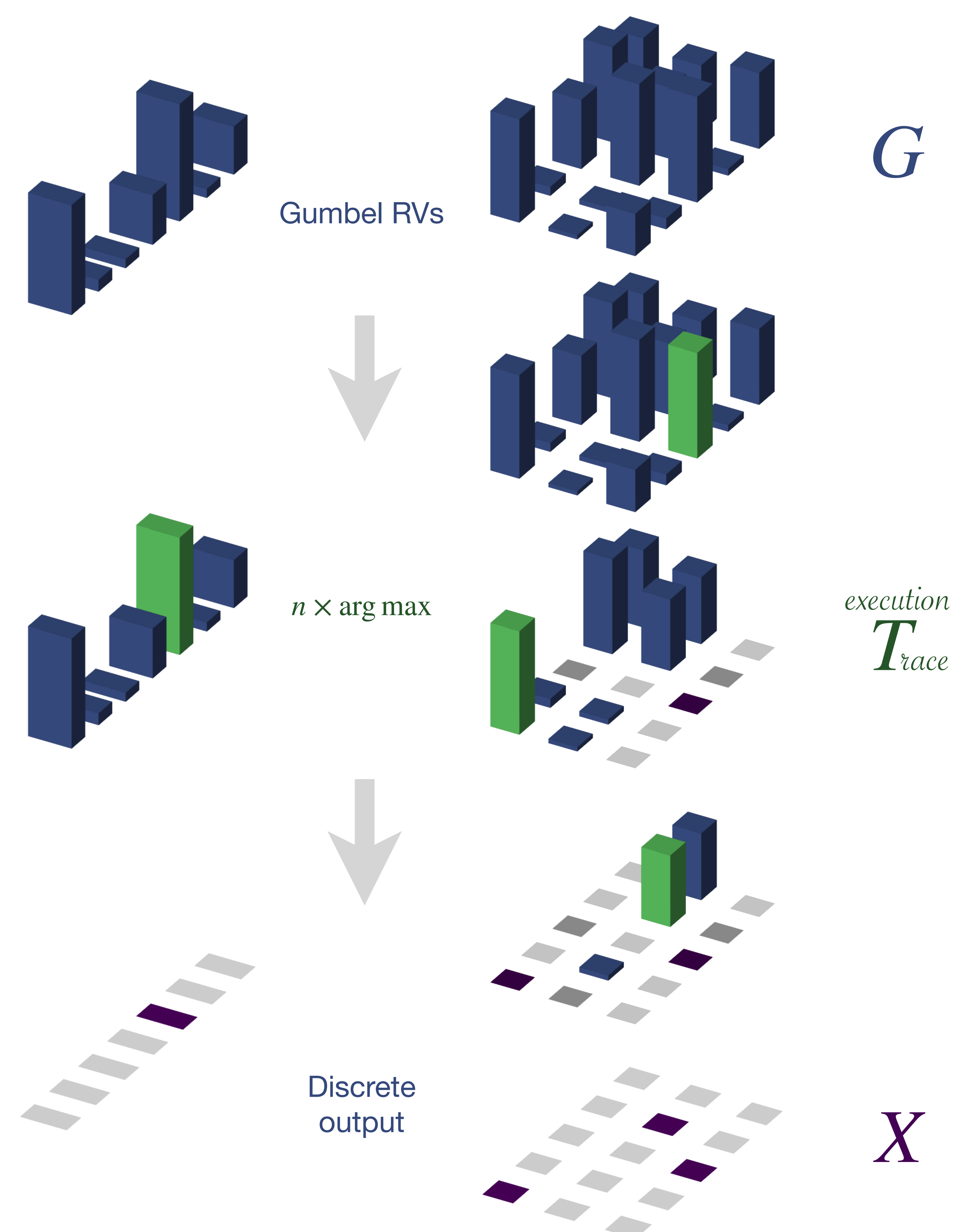
The Gumbel-Max Trick

The trick is a well-known sampling method for categorical distribution. In our work we use a less known fact about the Gumbel distribution. Specifically, non-maximum components again have the Gumbel distribution. Therefore we can reapply the trick to the non-maximum components.



Recursive Gumbel-Max Trick

Illustrated with Categorical Variables and Matchings



Input: Gumbel random variables $G \in \mathbb{R}^{n \times m}$

Output: Matching $X \in \{0,1\}^{n \times m}$

if $\min(n, m) = 0$:

return

$(i, j) \leftarrow \arg\max G$ // Apply the Gumbel-max trick

$G' \leftarrow$ cross out the i-th column and the j-th row of G

$X' \leftarrow$ **recurse** to construct matching for G' // G' is again Gumbel RVs

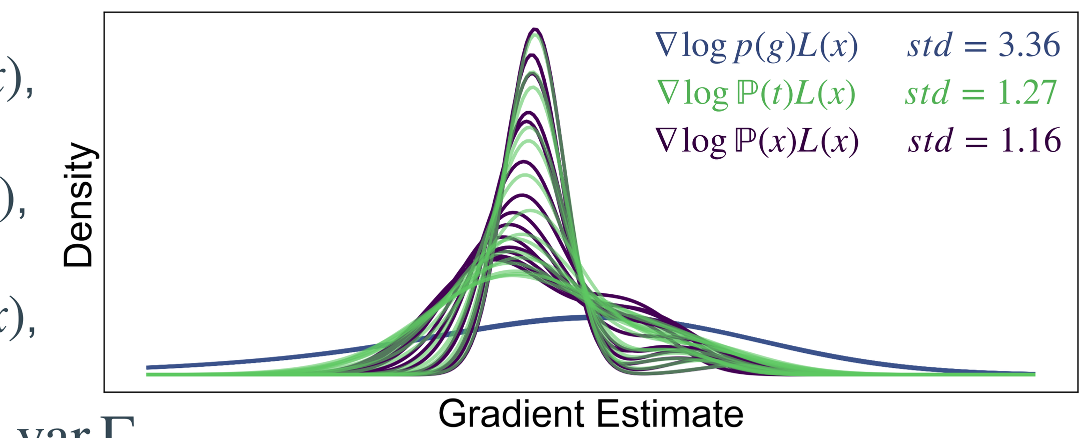
$X \leftarrow$ insert i-th column and j-th row into X', set $X_{ij} = 1$

return X

Learning with Score Function Estimates

Input G, execution trace T and output X yield three estimates for $\nabla \mathbb{E}\, L(X)$:

- $\Gamma_G = \nabla \log p(g)\ L(x)$,
- $\Gamma_T = \nabla \log \mathbb{P}(t)\ L(x)$,
- $\Gamma_X = \nabla \log \mathbb{P}(x)\ L(x)$,
- $\operatorname{var} \Gamma_G \geq \operatorname{var} \Gamma_T \geq \operatorname{var} \Gamma_X$.



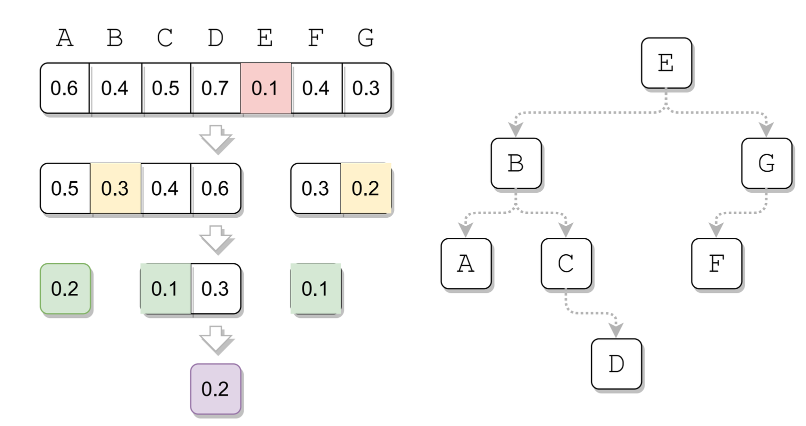
Γ_X is computationally heavy; we use Γ_T along with control variates

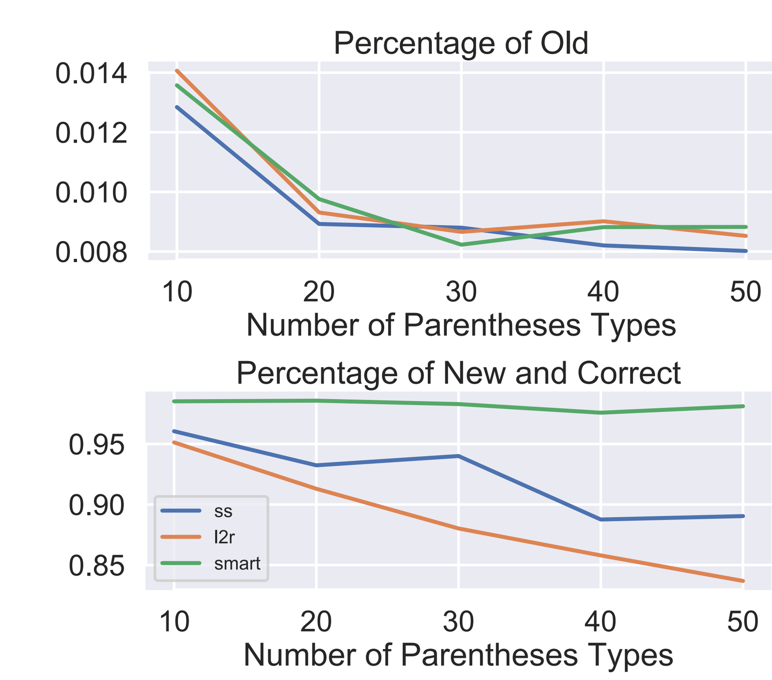
Applications

Comparison with Stochastic Softmax Tricks (SST) on ListOPS

Estimator	Accuracy		Precision		Recall	
	mean ± std	max	mean ± std	max	mean ± std	max
SST (Our Impl.)	78.42 ± 8.14	**93.78**	56.84 ± 20.08	**82.40**	30.18 ± 19.10	73.11
E-REINFORCE+	60.25 ± 2.29	64.47	40.87 ± 6.90	45.74	40.74 ± 6.93	45.46
T-REINFORCE+	**87.34 ± 3.00**	91.97	**77.93 ± 7.36**	79.65	**61.10 ± 14.11**	**79.65**
RELAX	79.60 ± 9.36	88.64	54.73 ± 17.48	75.27	53.61 ± 17.14	75.27

Non-monotonic generation of balanced parentheses





References

Paulus, Max, et al. "Gradient Estimation with Stochastic Softmax Tricks." NeurIPS 2020. 2020.

Nangia, Nikita, and Samuel Bowman. "ListOps: A Diagnostic Dataset for Latent Tree Learning." NAACLR Student Research Workshop. 2018.

Welleck, Sean, et al. "Non-monotonic sequential text generation." International Conference on Machine Learning. PMLR, 2019.

1 - HSE University, 2 - Artificial Intelligence Research Institute, 3 - Sber AI Lab, 4 - Skolkovo Institute of Science and Technology, 5 - Moscow Institute of Physics and Technology