Exact Bayesian Inference on Discrete Models via Probability Generating Functions: A Probabilistic Programming Approach

Fabian Zaiser Andrzej S. Murawski C.-H. Luke Ong

UNIVERSITY OF OXFORD

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

NeurIPS 2023

NEURAL INFORMATION PROCESSING SYSTEMS

Bayesian statistics

- Successful framework for reasoning under uncertainty
- Bayes’ law: prior beliefs & observations → posterior beliefs
- Inferring posterior distributions is a key challenge
- Analytical solutions are hard to find
- Approximate methods used in practice
  - Markov chain Monte Carlo (MCMC)
  - Variational Inference (VI)

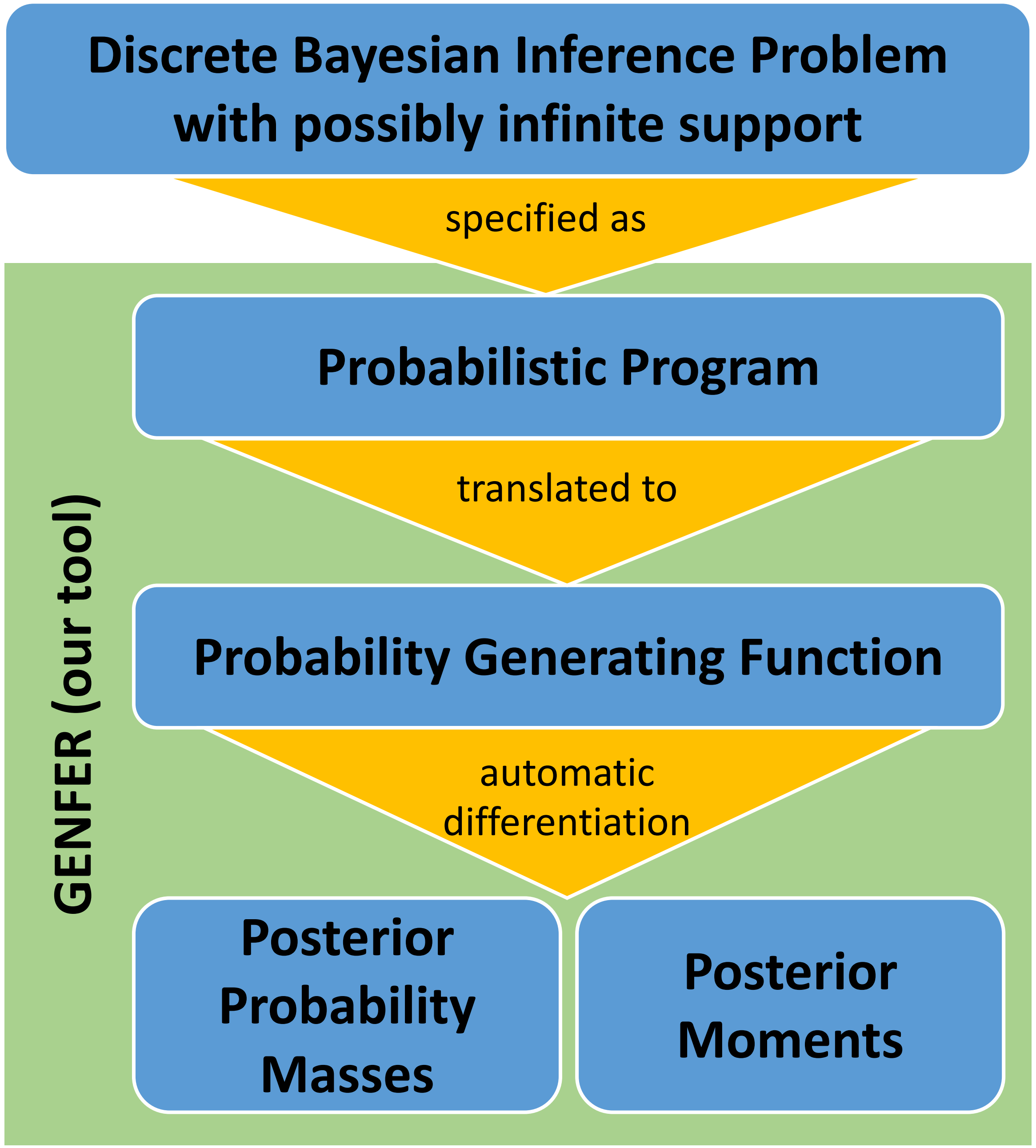
Our work:

Exact inference is possible for many discrete models!

Contributions

Exact inference is possible for a large class of **discrete** models even with **infinite support**

- in particular, **time series** models of **count data**
- **competitive with Monte-Carlo** methods on a range of benchmarks
- **often faster** than **existing exact methods** (when those apply)



Example: Animal population

- You're a biologist trying to estimate the size of an animal population
- You have a Poisson(20) prior for the population size
- You have a chance of 10% to see each animal
- You observe 2 animals

```
X ~ Poisson(20);
Y ~ Binomial(X, 0.1);
observe Y = 2;
```

$$P[X = x \mid Y = 2] = \frac{P[X = x] \cdot P[Y = 2 \mid X = x]}{P[Y = 2]}$$

Infinite support

```
X ~ Poisson(20);
Y ~ Binomial(X, 0.1);
observe Y = 2
```

$$\mathbb{P}[Y = 2] = \sum_{x=0}^{\infty} \mathbb{P}[\,Y = 2 \mid X = x\,] \cdot \mathbb{P}[X = x]$$

$$= \sum_{x=0}^{\infty} \mathrm{Binomial}(2; x, 0.1) \cdot \mathrm{Poisson}(x; 20)$$

Exact Inference Tools

```
X ~ Poisson(20);
Y ~ Binomial(X, 0.1);
observe Y = 2
```

Dice [Holtzen et al. 2020]

❌ only supports finite discrete distributions

SPPL [Saad et al. 2021]

❌ parameters of distributions must have finite support

PSI [Gehr et al. 2016]

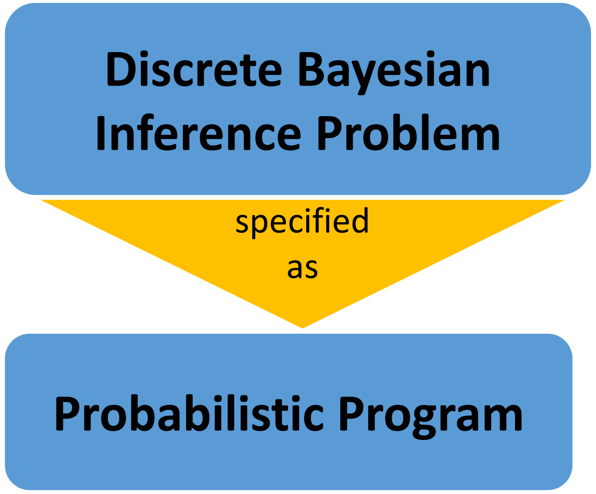
❌ outputs a symbolic expression with infinite sums

Probability Generating Functions

- **Generating function** of a random variable X is $G(t) := \mathbb{E}[t^X]$
- **Discrete** case: $G(t) = \sum_{k=1}^{\infty} \mathbb{P}[X = k] \cdot t^k$
- **Multivariate** case: $G(t_1, \dots, t_n) := \mathbb{E}[t_1^{X_1} \cdots t_n^{X_n}]$
- **Closed form** for many distributions and operations
  - Marginalizing out X_i: substituting 1 for t_i
  - Bayes' rule: $\frac{G(t_1,\dots,t_n)}{G(1,\dots,1)}$

Distribution	Generating function
$\text{Binomial}(n, p)$	$(1 - p + pt)^n$
$\text{Poisson}(\lambda)$	$e^{\lambda(t-1)}$
$\text{Binomial}(X, p)$	$G(1 - p + pt)$

Probabilistic Programming Language



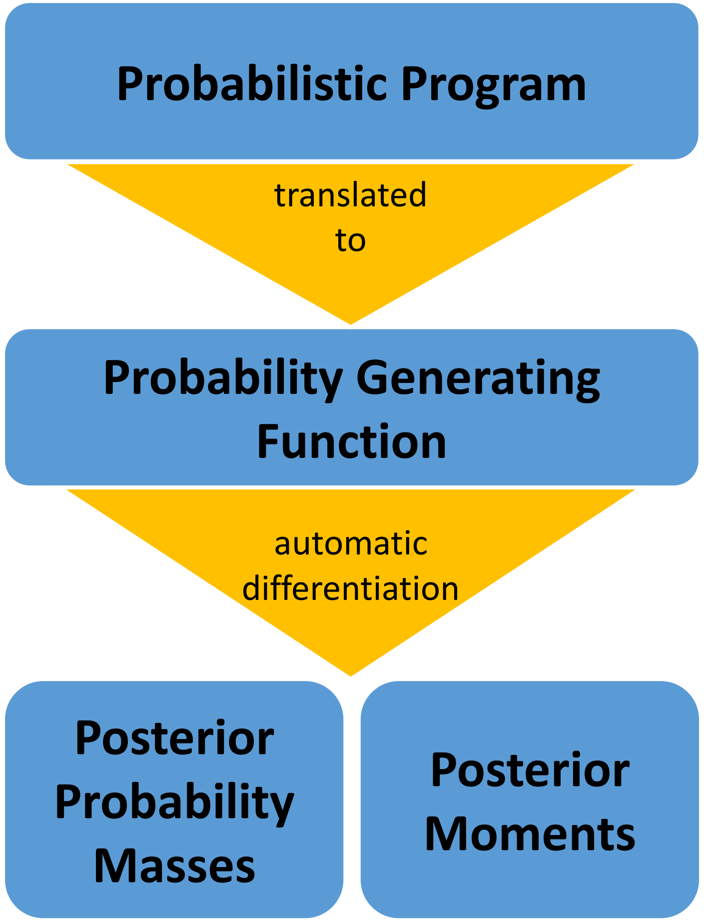
Flexible model specification:

- continuous & discrete priors
- stochastic branching
- affine transformations
- discrete observations

```
population ~ Poisson(100);
disaster ~ Bernoulli(0.1);

if disaster = 1 {
  population ~ Binomial(population, 0.8);
} else {
  offspring ~ Poisson(10);
  population += offspring;
}

observe 9 ~ Binomial(population, 0.1);

return population;
```

Translation to Generating Functions

- **Generating function represents the distribution of the program variables**
- Start with $G(t_1, \ldots, t_n) = 1$
- Each program statement **transforms** the generating function
- Our programming language ensures a **closed form** for the generating function
- Can extract probability **masses**: $\mathbb{P}[X = k] = \frac{G^{(k)}(0)}{k!}$
- Can extract **moments**: $\mathbb{E}[X] = G'(1)$



Running time

- Observing 100 corresponds to computing a 100th derivative!
- Symbolic computation would grow exponentially
- Instead: only evaluate the derivatives
- → **automatic differentiation via Taylor polynomials**
- Running time is $O(s \cdot d^{n+3})$ where
  - s: #statements in the program
  - d: sum of all observed values
  - n: #variables in the program

Limitations

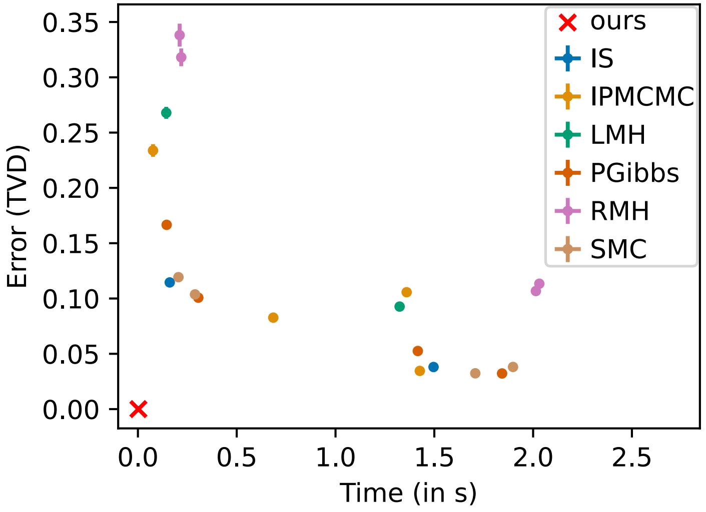
- Performance
  - Exact Bayesian inference is PSPACE-hard already for finite distributions
  - Running time of our method is exponential in #variables
  - But #variables can often be kept low by re-using variables (e.g. time series models)
- Expressiveness of the programming language
  - only affine operations supported
  - not all distributions support variables as parameters
  - no continuous observations (only continuous priors)
- No posterior densities (only posterior masses & moments)

Comparison with Exact Inference Methods

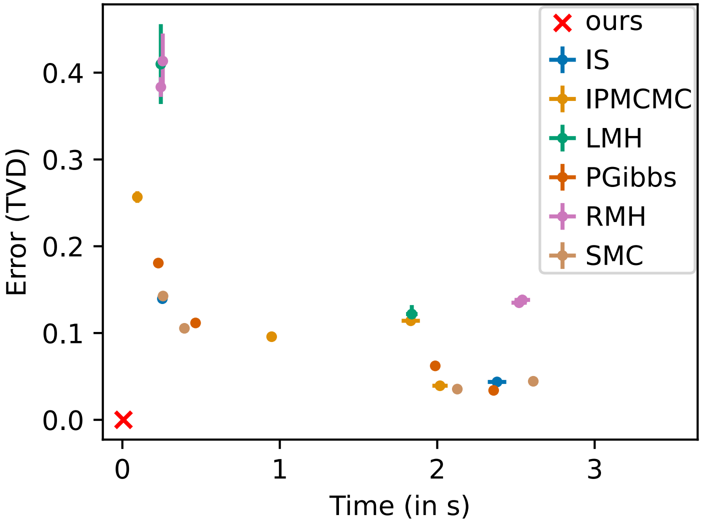
Benchmarks with bounded support

Tool	Genfer (FP)	Dice (FP)	Genfer ($\mathbb{Q}$)	Dice ($\mathbb{Q}$)	Prodigy	PSI
alarm (F)	**0.0005s**	0.0067s	**0.0012s**	0.0066s	0.011s	0.0053s
clickGraph (C)	**0.11s**	unsupported	**3.4s**	unsupported	unsupported	46s
clinicalTrial (C)	**150s**	unsupported	**1117s**	unsupported	unsupported	timeout
clinicalTrial2 (C)	**0.0024s**	unsupported	**0.031s**	unsupported	unsupported	0.46s
digitRecognition (F)	**0.021s**	0.83s	**0.11s**	2.7s	31s	146s
evidence1 (F)	**0.0002s**	0.0057s	**0.0003s**	0.0056s	0.0030s	0.0016s
evidence2 (F)	**0.0002s**	0.0056s	**0.0004s**	0.0057s	0.0032s	0.0018s
grass (F)	**0.0008s**	0.0067s	**0.0044s**	0.0067s	0.019s	0.014s
murderMystery (F)	**0.0002s**	0.0055s	**0.0003s**	0.0057s	0.0028s	0.0021s
noisyOr (F)	**0.0016s**	0.0085s	0.019s	**0.0088s**	0.21s	0.055s
twoCoins (F)	**0.0002s**	0.0054s	**0.0003s**	0.0057s	0.0032s	0.0017s

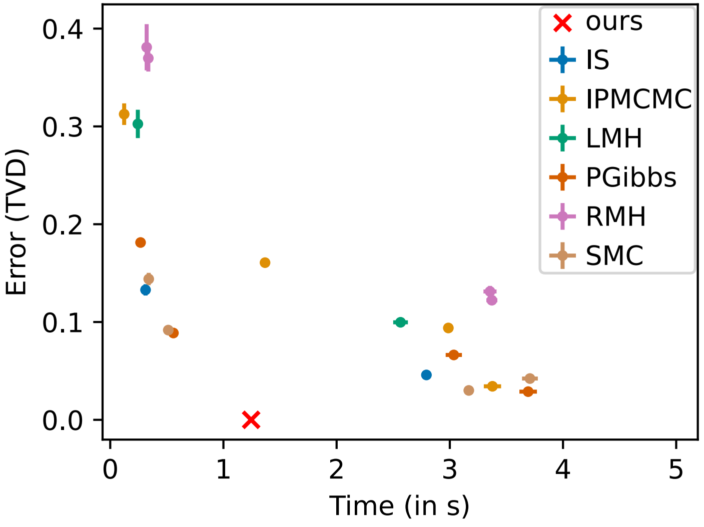
Comparison with Monte-Carlo Inference



Original Population Model



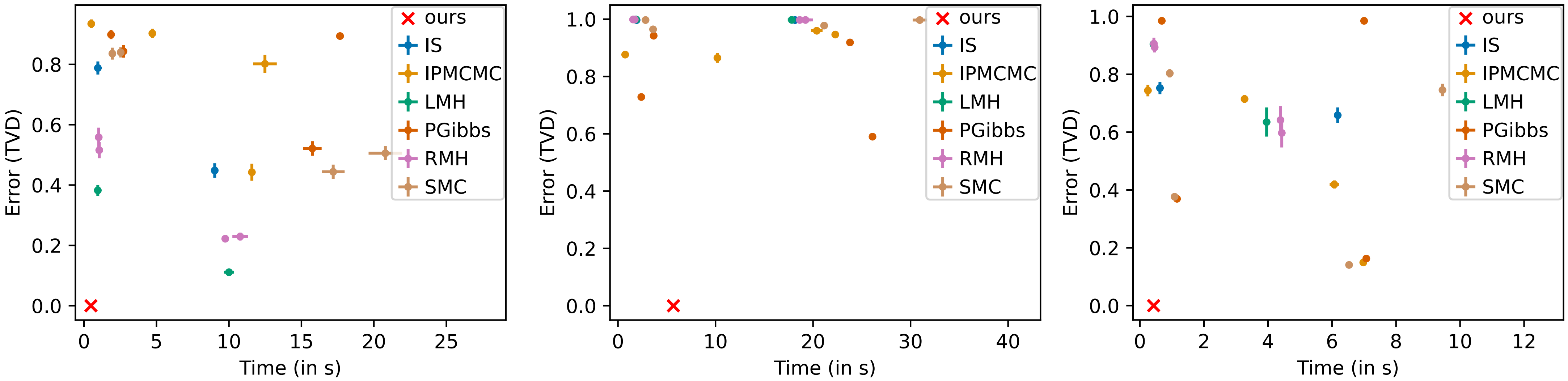
Modified Population Model



Two-type population model

Based on time series models with infinite support
in Winner et al. (NeurIPS 2016)

Comparison with Monte-Carlo Inference 2



Switchpoint Model
(conditioned on 109 data points)

Mixture Model
(conditioned on 109 data points)

Hidden Markov Model
(conditioned on 30 data points)

No exact solutions known before!

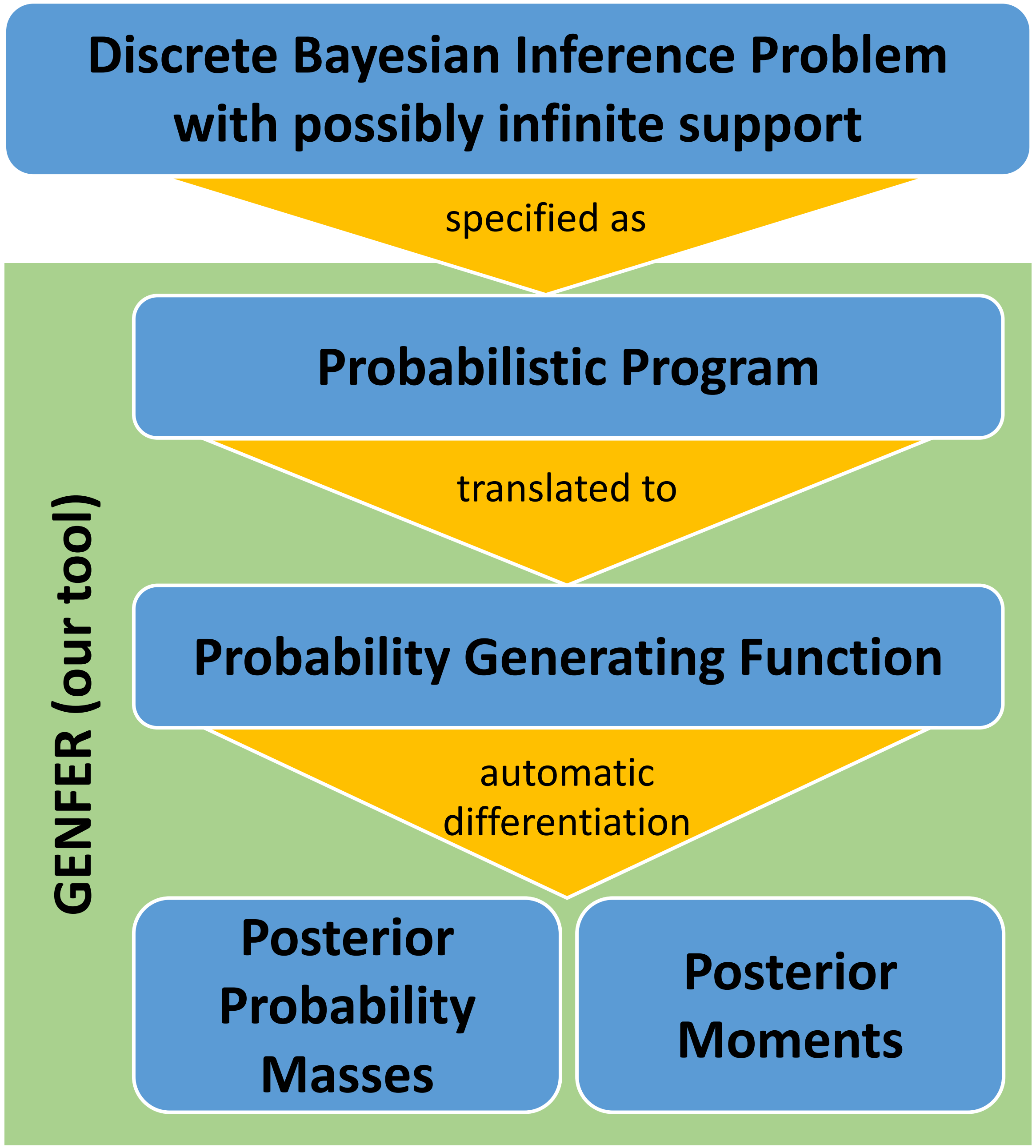
Conclusion

New framework for **exact Bayesian inference**

- **discrete models** even with **infinite support**
- **competitive performance** on a range of models
- **automated** in *Genfer*

Paper:

GitHub:



Backup Slides

Related Work on Generating Functions

	general models?	conditioning?	real-world examples?
Winner et al. (NeurIPS 2016)	❌	✅	✅
Winner et al. (ICML 2017)	❌	✅	✅
Klinkenberg et al. (LOPSTR 2020)	✅	❌	❌
Chen et al. (CAV 2022)	✅	❌	❌
Klinkenberg et al. (arXiv 2023)	✅	✅	❌
Our work	✅	✅	✅

Generating Functions: Example

$$1 \xrightarrow{X\sim\text{Poisson}(20)} e^{20(x-1)}$$
$$\xrightarrow{Y\sim\text{Binomial}(X,\,0.1)} e^{20(x(0.1y+0.9)-1)}$$
$$\xrightarrow{\text{observe } Y=2} 2x^2y^2e^{18x-20}$$
$$\xrightarrow{\text{normalize}} x^2y^2e^{18(x-1)} =: G(x,y)$$

```
X ~ Poisson(20);
Y ~ Binomial(X, 0.1);
observe Y = 2
```

Posterior probability: $\mathbb{P}[X = 10] = \frac{1}{10!}\frac{\partial^{10} G(0,1)}{\partial x^{10}} = 991796451840e^{-18}$

Posterior expectation: $\mathbb{E}[X] = \frac{\partial G(1,1)}{\partial x} = 20$