Graph Random Features for Scalable Gaussian Processes

Matthew Zhang[1], Jihao Andreas Lin[1,2], Krzysztof Choromanski[3], Adrian Weller[1,4], Richard E. Turner[1,4], Isaac Reid[1,3]

You can efficiently **approximate** the graph node kernel by **sampling random walks**, which unlocks **scalable Bayesian optimisation** on graphs with **>1M modes** on a **single computer chip**.

Graph random features

1. Many graph node kernels are expressed as functions of a **weighted adjacency matrix**

$$\mathbf{K}_\alpha(\mathbf{W}) = \sum_{r=0}^{\infty} \alpha_r \mathbf{W}^r, \quad \alpha_r \in \mathbb{R} \ \ \forall r \in (0, 1, ..., \infty).$$

2. Powers of $\mathbf{W}$ count random walks, so we can estimate this by importance sampling (a.k.a. 'graph random features', or **GRFs**)

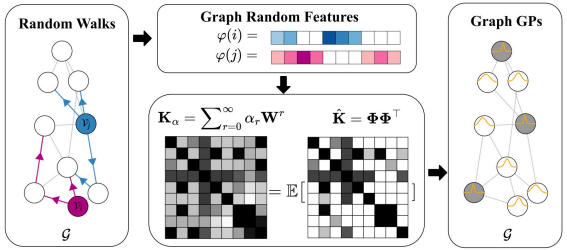

Fig 1. **Schematic.** *GRFs for Bayesian Inference schematic*

Efficient GPs on graphs

1. Under mild assumptions on the graph, the kernel estimate $\hat{\mathbf{K}} = \mathbf{\Phi}\mathbf{\Phi}^\top$ is provably **sparse** and has a **bounded condition number**.

2. This lets us **estimate** quantities involving the kernel inverse like the posterior mean and variance

$$m_{|\boldsymbol{y}}(x) = m(x) + k(x, \boldsymbol{x})\big[k(\boldsymbol{x}, \boldsymbol{x}) + \sigma_n^2 \mathbf{I}\big]^{-1}(\boldsymbol{y} - m(\boldsymbol{x})),$$

$$k_{|\boldsymbol{y}}(x, x') = k(x, x') - k(x, \boldsymbol{x})\big[k(\boldsymbol{x}, \boldsymbol{x}) + \sigma_n^2 \mathbf{I}\big]^{-1} k(\boldsymbol{x}, x').$$

very efficiently using iterative linear system solvers, e.g. **conjugate gradient methods**.

5. Combining with **pathwise conditioning**

$$\boldsymbol{g}_{|\boldsymbol{y}}(\cdot) = \boldsymbol{g}(\cdot) + \hat{\mathbf{K}}_{(\cdot)\boldsymbol{x}}\big(\hat{\mathbf{K}}_{\boldsymbol{x}\boldsymbol{x}} + \sigma_n^2 \mathbf{I}\big)^{-1}(\boldsymbol{y} - (\boldsymbol{g}(\boldsymbol{x}) + \varepsilon)).$$

we can efficiently approximate **inverse kernel-vector products,** unlocking Bayesian optimisation on massive graphs in $\mathcal{O}\big(N^{3/2}\big)$ time.

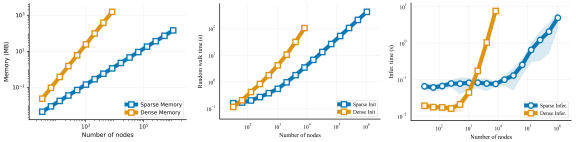

Fig 2. **Scaling curves.** *GRF-GPs have sub-quadratic time scaling and linear memory scaling. From left to right: the scaling of memory, kernel computation (random walk sampling) time and inference time respectively.*

Applications and experiments

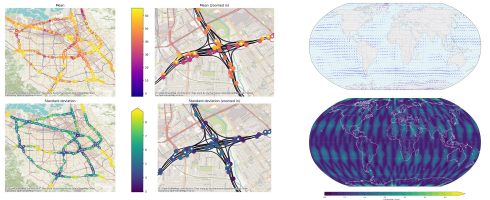

Fig 3. **Regression tasks: traffic prediction and wind interpolation.** *GRFs for uncertainty-aware regression tasks real world graph datasets. Left: traffic speed prediction. Right: Wind velocity interpolation.*

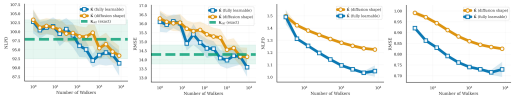

Fig 4. **GRFs surpass the exact diffusion GPs with ~500 walkers per node.** *NLPD and RMSE versus the number of random walkers. From left to right: traffic NLPD, RMSE, wind NLPD, RMSE.*

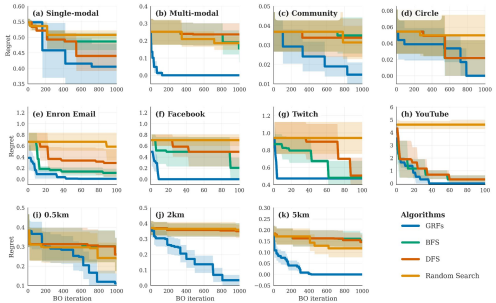

Fig 5. **Bayesian optimisation.** *GRFs beat other scalable baselines for Bayesian optimisation tasks on graphs datasets with over 1M nodes.*

[1] *University of Cambridge,* [2] *Max Planck Institute for Intelligent Systems,* [3] *Google DeepMind,* [4] *Alan Turing Institute*