GRAND: Graph Neural Diffusion Supplementary Material

Benjamin P. Chamberlain* 1 James Rowbottom* 1 Maria Gorinova 1 Stefan Webb 1 Emanuele Rossi 1 Michael M. Bronstein 1 2

1. Datasets
The statistics for the largest connected components of the experimental datasets are given in Table 1.

2. Diffusivity Formulations
GRAND can use any right stochastic attention matrix. We performed experiments with the multiheaded Bahdanau formulation (Bahdanau et al., 2014) of attention, which has previously been applied to graphs in (Veličković et al., 2018)

where $W$ and $a$ are learned and $\| \$ is the concatenation operator. However, for all datasets, the scaled dot product attention performed better. This may be because GAT relies on dropout. Dropout performs poorly inside adaptive timestep numerical ODE solvers as the stochasticity in the forward pass drives $\tau \rightarrow 0$.

3. Full Training Objective
The full training program optimises cross entropy loss

$$L(Y, T) = H(Y, T) = \sum_{i=1}^{n} t_i^T \log y_i$$ \hspace{1cm} (2)

where $t_i \in \mathbb{R}^{d_{\text{true}}}$ is the one-hot truth vector of the $i^{th}$ node with prediction

$$y_i = \psi(x_i(T))$$ \hspace{1cm} (3)

where $\psi : \mathbb{R}^d \rightarrow \mathbb{R}^{d_{\text{true}}}$ is a linear layer decoder of the terminal value of the evolutionary PDE

$$y_i = D x_i(T) + b_d$$ \hspace{1cm} (4)

$$= D \left( \phi(X_{in}) + \int_0^T \frac{\partial X(t)}{\partial t} dt \right) + b_d$$ \hspace{1cm} (5)

with the initial condition given by the linear layer encoder $\phi : \mathbb{R}^{d_{in}} \rightarrow \mathbb{R}^d$ of the input data

$$y_i = \left( D \left( E(X_{in}) + b_e + \int_0^T \frac{\partial X(t)}{\partial t} dt \right) + b_d \right)_i$$

and $f(X(t), t, \theta) = \frac{\partial X(t)}{\partial t}$ is the system dynamics that we wish to learn. For the nonlinear version of GRAND this is

$$f = \frac{\partial}{\partial t} X(t) = (A(X(t)) - I)X(t) = \tilde{A}(X(t))X(t)$$

4. Stability

4.1. Stability of linear ODE
In the main paper we reported the linear GRAND $\dot{x} = \tilde{A}x$ has solution

$$x(t) = x(0)e^{\tilde{A}t}.$$ \hspace{1cm} (7)

As $\tilde{A}$ is not diagonal this matrix exponential is not analytically recoverable. Performing eigenvalue decomposition the solution is

$$x(t) = \bar{T}e^{Dt}\bar{T}^{-1}x(0).$$ \hspace{1cm} (8)

Assuming $\bar{T}^{-1}$ exists, $\bar{T}$ has full rank and both are bounded, the test equation becomes

$$y(t) = e^{D_{\text{test}}t}y(0)$$ \hspace{1cm} (9)

where $y(t) = \bar{T}x(t)$. If $x(t)$ and $\tilde{x}(t)$ are two solutions of the ODE then their projections in eigenspace are $y(t)$ and $\hat{y}(t)$. For each node $i$:

$$|y_i(t) - \hat{y}_i(t)| = \left| (y_i(0) - \hat{y}_i(0))e^{\tilde{A}_i t} \right|$$ \hspace{1cm} (10)

$$= |y_i(0) - \hat{y}_i(0)|e^{Re(\tilde{\lambda}_i) t}$$ \hspace{1cm} (11)

for this to converge as $t \rightarrow \infty$ we require $Re(\tilde{\lambda}_i) \leq 0 \forall i$. As $A$ is right stochastic the eigenvalues of $\tilde{A} = A - I$ satisfy this property.
5. Numerical Schemes

5.1. Proof of theorem 1: Stability of explicit Euler

For linear GRAND with an Euler numerical integrator

\[ \mathbf{x}^{(t+1)} = \left( I + \tau \mathbf{A} \mathbf{x}^{(t)} \right) \mathbf{x}^{(t)} = \mathbf{Q}^{(t)} \mathbf{x}^{(t)}. \]  

We require that the amplification factor \( ||\mathbf{Q}^{(t)}|| < 1 \). It is sufficient to show that \( \mathbf{Q}^{(t)} \) is a right stochastic matrix, which has the property that its spectral radius \( \lambda_{\text{max}} \leq 1 \). \( \mathbf{Q} \) is right stochastic if

1. \( \sum_{j=1}^{N} q_{ij} = 1 \)
2. \( q_{ij} > 0 \) \( \forall i,j \)

as \( \mathbf{A} \) is right stochastic \( \sum_{j} I_{ij} + \tau (A_{ij} - I_{ij}) = 1 \) proving 1. As \( a_{ij} = q_{ij} \) for \( i \neq j \), to prove 2) it remains to show that \( 1 + \tau (a_{ii} - 1) > 0 \iff \tau < 1 \).

5.2. Proof of theorem 2: Implicit methods

For implicit Euler

\[
\dot{x}_n = \frac{x_n - x_{n-1}}{\tau} = f(x_n, t_n) \tag{14}
\]
\[
x_n = \tau f(x_n, t_n) + x_{n-1}, \tag{15}
\]

incrementing the indices gives

\[
x_{n+1} = \tau f(x_{n+1}, t_{n+1}) + x_{n} \tag{16}
\]

and now, unlike the explicit case, \( x_{n+1} \) now appears on both sides of the equation. If \( f \) is linear

\[
x_{n+1} = \tau \bar{A} x_{n+1} + x_{n} \tag{17}
\]
\[
x_{n+1} = (I - \tau \bar{A})^{-1} x_{n} = B^{-1} x_{n} = Q x_{n}, \tag{18}
\]

and the matrix \( B \) must be inverted. The inverse exists as \( B \) is diagonally dominant

\[
I_{ii} - \tau (A_{ii} - I_{ii}) > \tau \sum_{j \neq i} A_{ij} = \tau (1 - A_{ii}) \tag{19}
\]

By considering the action of \( B \) on \( w = (1, \ldots, 1)^T \) it is clear that \( B w = w \implies Q w = w \implies \sum_{j} Q_{ij} = 1 \). As \( B \) is diagonally dominant it is irreducible and satisfies \( B_{ij} \leq 0 \) \( i \neq j \) and \( B_{ii} > 0 \) giving \( Q_{ij} > 0 \) \( \forall i,j \) (Varga, 1999) and \( Q \) is a Markov matrix with spectral radius bounded by unity and the implicit scheme is stable for all choices of \( \tau \).

6. General Multistep Methods

A general multistep method (combining both implicit and explicit methods) can be written as

\[
x_{n+1} + \sum_{i=1}^{s} \alpha_{i} x_{n+1-i} = \tau \sum_{i=0}^{s} \beta_{i} f_{n+1-i}, \tag{20}
\]

where \( f = \dot{x} \). If \( \beta_{0} = 0 \) then \( x_{n+1} \) only depends on terms up to \( n \) and the method is explicit.

6.1. Order

The order of a method gives the approximation error in terms of a Taylor series expansion. If \( p \) is the order, then the error is a single step \( \propto \tau^{p+1} \) and the error in the entire interval \( \propto \tau^{p} \). In practice the order of a numerical method can be determined by measuring how the error changes with step size for a known integral.

6.2. Butcher Tableau

The set of coefficients for each multi step method are given by the Butcher Tableau. The simple case of forward Euler has \( \alpha_{1} = -1, \beta_{1} = 1 \) with all other terms zero.

There is a law of diminishing return that relates the minimum number of function evaluations and the order of a higher order Runge-Kutta solver. Table 2 shows why the Runge-Kutta 4 method (RK4) is often regarded as the optimal trade-off between speed and accuracy for multi step solvers.

6.3. Runge-Kutta 4

For all experiments we find that Runge-Kutta 4 (or its adaptive step size variants) outperforms lower order methods. The Runge-Kutta 4 method follows the schema: if

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Type</th>
<th>Classes</th>
<th>Features</th>
<th>Nodes</th>
<th>Edges</th>
<th>Label rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cora</td>
<td>citation</td>
<td>7</td>
<td>1433</td>
<td>2485</td>
<td>5069</td>
<td>0.056</td>
</tr>
<tr>
<td>Citeseer</td>
<td>citation</td>
<td>6</td>
<td>3703</td>
<td>2120</td>
<td>3679</td>
<td>0.057</td>
</tr>
<tr>
<td>PubMed</td>
<td>citation</td>
<td>3</td>
<td>500</td>
<td>19717</td>
<td>44324</td>
<td>0.003</td>
</tr>
<tr>
<td>Coauthor CS</td>
<td>co-author</td>
<td>15</td>
<td>6805</td>
<td>18333</td>
<td>81894</td>
<td>0.016</td>
</tr>
<tr>
<td>Computers</td>
<td>co-purchase</td>
<td>10</td>
<td>767</td>
<td>13381</td>
<td>245778</td>
<td>0.015</td>
</tr>
<tr>
<td>Photos</td>
<td>co-purchase</td>
<td>8</td>
<td>745</td>
<td>7487</td>
<td>119043</td>
<td>0.021</td>
</tr>
<tr>
<td>OGB-Arxiv</td>
<td>citation</td>
<td>40</td>
<td>128</td>
<td>169343</td>
<td>1166243</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 1. Dataset Statistics
Table 2. Function evaluations grow super-linearly with order after 4.

\[ f(x, t) = \bar{A}(x_t) x_t \]

\[ x^{(t+1)} = x^{(t)} + \frac{1}{6} \tau (k_1 + 2k_2 + 2k_3 + k_4) \]  
(21)

\[ k_1 = f(x_t, t) \]  
(22)

\[ k_2 = f(x_t + \tau k_1/2, t + \tau/2) \]  
(23)

\[ k_3 = f(x_t + \tau k_2/2, t + \tau/2) \]  
(24)

\[ k_4 = f(x_t + \tau k_3, t + \tau) \]  
(25)

6.4. Adaptive Step Size

Adaptive step size solvers estimate the error in \( x_{n+1} \), which is compared to an error tolerance. The error is estimated by comparing two methods, one with order \( p \) and one with order \( p - 1 \). They are interwoven, i.e., they have common intermediate steps. As a result, estimating the error has little or negligible computational cost compared to a step with the higher-order method.

\[ x^*_{n+1} = x_n + \tau \sum_{i=1}^{s} b_i^* k_i \]  
(26)

where \( k_i \) are the same as for the higher-order method. Then the error is

\[ e_{n+1} = x_{n+1} - x^*_{n+1} = \tau \sum_{i=1}^{s} (b_i - b_i^*) k_i. \]  
(27)

The time step is increased if the error is below tolerance and decreased otherwise.

7. Adaptive step size implementation details

Most results presented used the adaptive step size solver Dormand-Prince5. Key to getting this to work well is setting appropriate tolerances for the step size. Adaptive step size ODE solvers require two tolerance parameters; the relative tolerance \( rtol \) and the absolute \( atol \). Both are used to assess the new step size

\[ etol = atol + rtol \times \max(|x_0|, |x_1|), \]  
(28)

where \( x_0 \) and \( x_1 \) are successive estimations of the new state. Dupont et al. (2019) speculate that ResNets can learn a richer class of functions than ODEs because “the error arising from discrete steps allows trajectories to cross”. We find that increasing the estimation error is also helpful when learning continuous diffusion functions and use value of \( rtol \) and \( atol \) that are \( \times10^{-12} \) and \( atol = ts^{−6} \).

When using the adjoint method to backpropagate derivatives, two separate ODEs are being solved. This requires separate tolerance scales, which may differ: the forward pass tolerance, \( ts \), controls for how close the approximated ODE solution is compared to the true solution, while the backward pass tolerance, \( ts_{adj} \), controls the accuracy of the computed gradient. The hyperparameter search includes both \( ts \) and \( ts_{adj} \).

References


