RBF-generated Finite Differences for Elliptic PDEs on Multiple GPUs

Evan F. Bollig[1] Gordon Erlebacher[1] Natasha Flyer[2]

[1]Dept. of Scientific Computing, Florida State University
[2]Institute for Mathematics Applied to Geosciences (IMAGe), NCAR

Introduction

Radial Basis Functions (RBFs) provide a powerful and elegant solution to calculate weights for generalized Finite Differences on arbitrary node distributions. Weights apply to stencils of scattered nodes (e.g., Figure 1) and result in a derivative approximation at the stencil center. High-order accuracy is easily achieved by increasing the number of nodes per stencil.

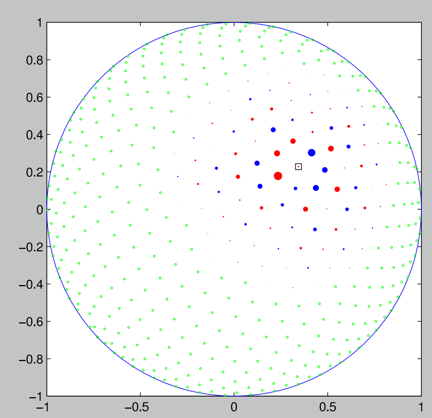


Figure 1 : A 75 node RBF-FD stencil with blue (negative) and red (positive) differentiation weights to approximate a derivative at the center (black square).

This effort extends previous work on a multi-CPU/GPU implementation of RBF-FD originally dedicated to explicit solutions of hyperbolic PDEs [1]. The addition of a GPU-based implicit solver for elliptic PDEs completes the necessary building blocks required for large-scale GPU solution of geophysical flows based entirely on the RBF-FD method.

RBF-FD Weights (for one n-node stencil centered at x_j)

$$\begin{pmatrix} \phi(\epsilon\|\mathbf{x}_1-\mathbf{x}_1\|) & \phi(\epsilon\|\mathbf{x}_1-\mathbf{x}_2\|) & \cdots & \phi(\epsilon\|\mathbf{x}_1-\mathbf{x}_n\|) & 1 \\ \phi(\epsilon\|\mathbf{x}_2-\mathbf{x}_1\|) & \phi(\epsilon\|\mathbf{x}_2-\mathbf{x}_2\|) & \cdots & \phi(\epsilon\|\mathbf{x}_2-\mathbf{x}_n\|) & 1 \\ \vdots & & \ddots & \vdots & \vdots \\ \phi(\epsilon\|\mathbf{x}_n-\mathbf{x}_1\|) & \phi(\epsilon\|\mathbf{x}_n-\mathbf{x}_2\|) & \cdots & \phi(\epsilon\|\mathbf{x}_n-\mathbf{x}_n\|) & 1 \\ 1 & 1 & \cdots & 1 & 0 \end{pmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \\ c_{n+1} \end{bmatrix} = \begin{bmatrix} \mathcal{L}\phi(\epsilon\|\mathbf{x}-\mathbf{x}_1\|)|_{\mathbf{x}=\mathbf{x}_j} \\ \mathcal{L}\phi(\epsilon\|\mathbf{x}-\mathbf{x}_2\|)|_{\mathbf{x}=\mathbf{x}_j} \\ \vdots \\ \mathcal{L}\phi(\epsilon\|\mathbf{x}-\mathbf{x}_n\|)|_{\mathbf{x}=\mathbf{x}_j} \\ 0 \end{bmatrix} \quad (1)$$

- ϕ is Gaussian RBF centered at x_k, $k=1,\dots,n$
- $\mathcal{L}$ is some differential operator (i.e., $\frac{\partial}{\partial x}$, $\frac{\partial}{\partial y}$, ∇^2, etc.); form multiple RHS system for efficiency
- Repeat this $n\times n$ system solve for all N stencils.

Governing Equation

Steady-state viscous Stokes flow on the surface of a sphere:

$$\nabla\cdot[\eta(\nabla\mathbf{u}+(\nabla\mathbf{u})^T)] + RaT\hat{r} = \nabla p$$
$$\nabla\cdot\mathbf{u} = 0,$$

Assume constant η (i.e., $\nabla\eta = 0$) to simplify test problem:

$$\begin{pmatrix} -\eta\nabla^2 & 0 & 0 & \frac{\partial}{\partial x_1} \\ 0 & -\eta\nabla^2 & 0 & \frac{\partial}{\partial x_2} \\ 0 & 0 & -\eta\nabla^2 & \frac{\partial}{\partial x_3} \\ \frac{\partial}{\partial x_1} & \frac{\partial}{\partial x_2} & \frac{\partial}{\partial x_3} & 0 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ p \end{pmatrix} = \frac{RaT}{\sqrt{x_1^2+x_2^2+x_3^2}} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ 0 \end{pmatrix}. \quad (2)$$

Simplifications for Development

- ∇^2 operator on the unit sphere:

$$\nabla^2 = \underbrace{\frac{1}{\hat{r}}\frac{\partial}{\partial\hat{r}}\left(\hat{r}^2\frac{\partial}{\partial\hat{r}}\right)}_{\text{radial}} + \underbrace{\frac{1}{\hat{r}^2}\Delta_S}_{\text{angular}} \equiv \Delta_S,$$

  - Diagonal block RBF-FD weight operator (i.e., RHS of Eq. 1)

$$\Delta_S = \frac{1}{4}\left[(4-r^2)\frac{\partial^2}{\partial r^2} + \frac{4-3r^2}{r}\frac{\partial}{\partial r}\right], \quad (3)$$

  where r is the Euclidean distance between stencil nodes and independent of coordinate system.

- $\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, \frac{\partial}{\partial x_3}$ must be constrained to the sphere via projection:

$$P_x = I - \mathbf{x}\mathbf{x}^T$$

  - Off-diagonal block operators:

$$P_x\frac{\partial}{\partial x_1} = (x_1\mathbf{x}^T\mathbf{x}_k - x_{1,k})\frac{1}{r}\frac{\partial}{\partial r}|_{\mathbf{x}=\mathbf{x}_j} \quad (4)$$
$$P_x\frac{\partial}{\partial x_2} = (x_2\mathbf{x}^T\mathbf{x}_k - x_{2,k})\frac{1}{r}\frac{\partial}{\partial r}|_{\mathbf{x}=\mathbf{x}_j} \quad (5)$$
$$P_x\frac{\partial}{\partial x_3} = (x_3\mathbf{x}^T\mathbf{x}_k - x_{3,k})\frac{1}{r}\frac{\partial}{\partial r}|_{\mathbf{x}=\mathbf{x}_j} \quad (6)$$

The Bane of RBF Methods: Choosing the Right Support

- Choice of ϵ determines accuracy of weights
- Trade-off: increase $\log_{10}\hat{\kappa}_A$ for accurate derivatives, worsen conditioning of system
- Contours change with stencil size (n) and node-distribution

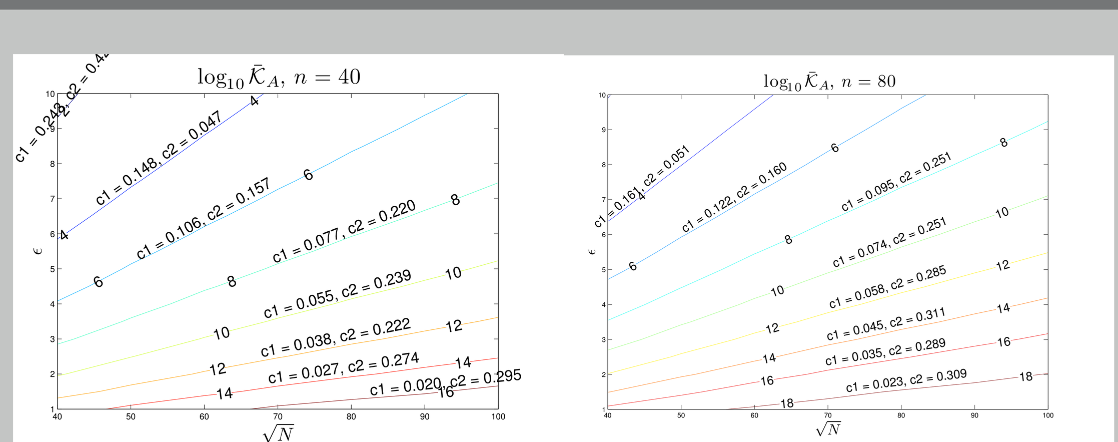
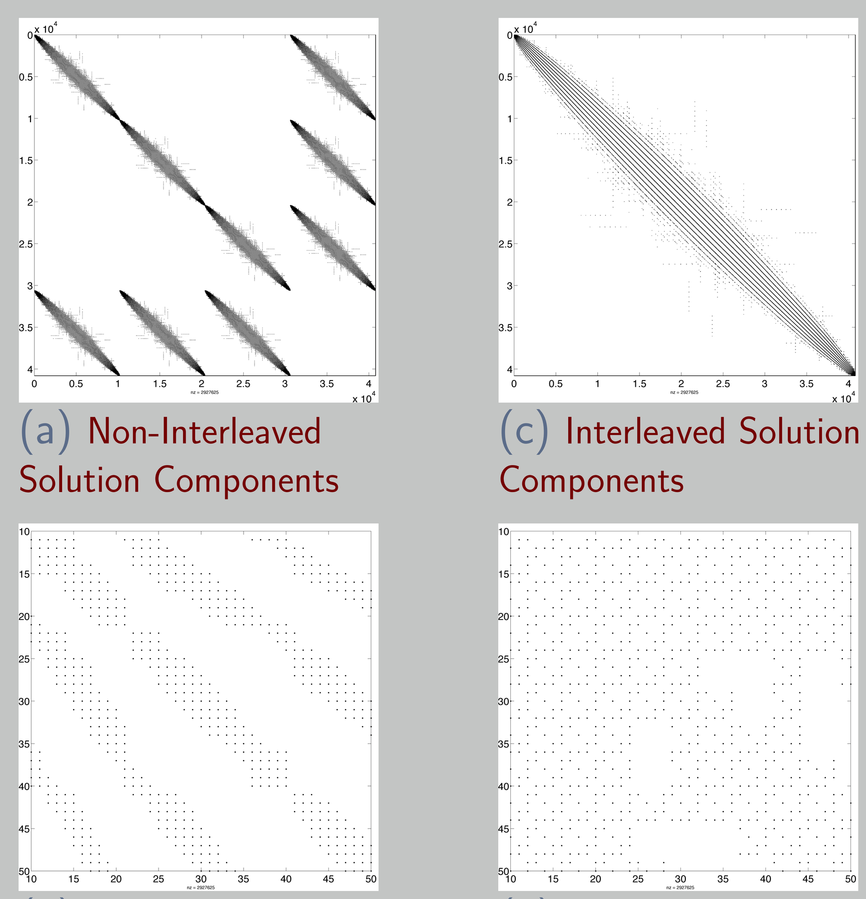


Figure 2 : Reliably choose ϵ given a condition number and number of nodes on the sphere: $\epsilon(N,\log_{10}\hat{\kappa}_A) = c_1(\log_{10}\hat{\kappa}_A)\sqrt{N} - c_2(\log_{10}\hat{\kappa}_A)$

GPU Matrix Ordering – Increase Memory Loads



(a) Non-Interleaved Solution Components
(c) Interleaved Solution Components
(b) Non-Interleaved Submatrix $(10:50)^2$
(d) Interleaved Submatrix $(10:50)^2$

Figure 3 : Sparsity pattern of linear system in Equation 2. Solution values are either non-interleaved and grouped by component (e.g., $(u_1,\cdots,u_N,v_1,\cdots,v_N,\cdots,p_1,\cdots,p_N)^T$) or interleaved (e.g., $(u_1,v_1,w_1,p_1,\cdots,u_N,v_N,w_N,p_N)^T$).

- Interleaving simplifies index management in domain decomposition
- Improve memory access for certain sparse storage formats

Decomposition/Communication Sets for Multi-GPU

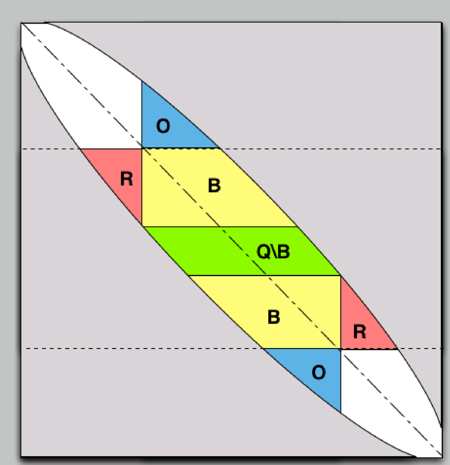


Figure 4 : Matrix decomposition for one GPU and the stencils (rows) involved in MPI data transfer

- One GPU is associated with every CPU
- Stencils reordered internally on each GPU: $\{Q\backslash B, B\backslash O, O, R\}$
  - Keep O and R contiguous for fast transfer between CPU and GPU

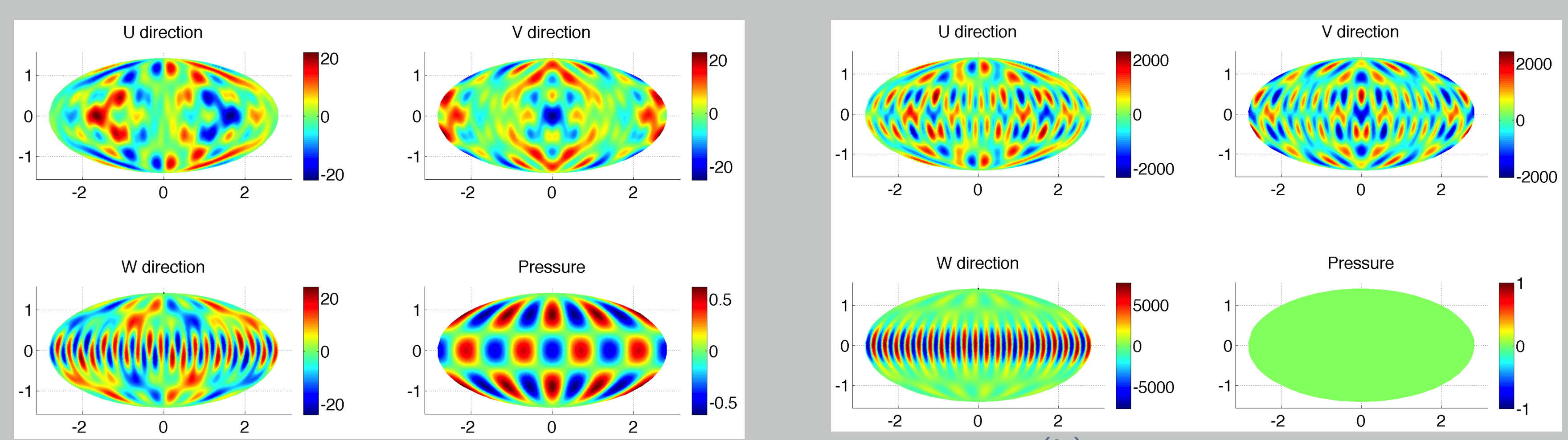
Manufacture Divergence-Free Fields

- For any function $g(\mathbf{x})$, $\mathbf{u} = Q_x P_x g(\mathbf{x})$ where Q_x is the curl projection:

$$Q_x = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}.$$

- Spherical Harmonics (Y_l^m) test case:

$$g(x) = 8Y_3^2 - 3Y_{10}^5 + Y_{20}^{20}$$
$$P = Y_6^4$$



(a) Solution $Q_x(g(x))$ with $g(x) = 8Y_3^2 - 3Y_{10}^5 + Y_{20}^{20}$
(b) Manufactured RHS

Implicit Solutions with Preconditioned GMRES

Algorithm 1 : Left-preconditioned GMRES(k)

```
while convergence == false do
    r_0 = M^{-1}(b - A x_0)                         ▷ MPI_Alltoallv
    β = ||r_0||_2                                   ▷ MPI_Allreduce
    v_1 = r_0/β
    for j = 1 to k do
        w_j = M^{-1} A v_j                          ▷ MPI_Alltoallv
        for i = 1 to j do
            h_{i,j} =< w_j, v_i >                   ▷ MPI_Allreduce
            w_j = w_j - h_{i,j} v_i
        end for
        h_{j+1,j} = ||w_j||_2                       ▷ MPI_Allreduce
        v_{j+1} = w_j/h_{j+1,j}
    end for
    Set V_k = [v_1, ..., v_k] and H̄_k = (h_{i,j})
    Solve: min_{y∈R^k} ||β e_1 - H̄_k y||_2
    x_k = x_0 + V_k y_k
    if ||M^{-1}(b - A x_k)||_2 < ε then
        convergence = true
    end if
    x_0 = x_k                                       ▷ MPI_Alltoallv
end while
```

- Sparse Matrix-Vector Multiply (SpMV) is true bottleneck in GMRES

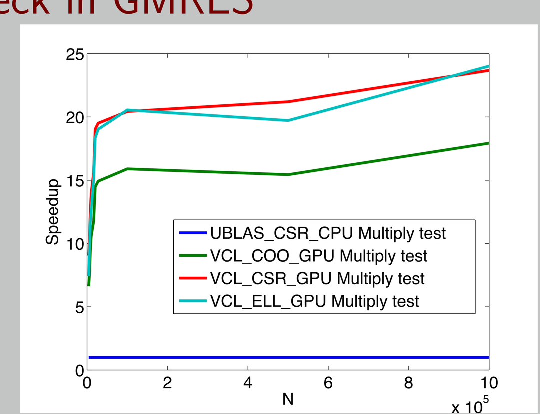


Figure 6 : The GPU accelerates the SpMV by up to 24x over the CPU

- RBF-FD systems are slow to converge
  - Investigating preconditioners
  - Accurate and convergent solutions may require stable algorithm for RBF-FD weight calculation

Acknowledgements

This work is supported by NSF awards DMS-#0934331 (FSU), DMS-#0934317 (NCAR) and ATM-#0602100 (NCAR). The following computing resources were utilized during this investigation: the FSU HPC Spear Cluster and the NSF Keeneland Cluster (OCI-0910735).

[1] Evan F. Bollig, Natasha Flyer, and Gordon Erlebacher. Solution to pdes using radial basis function finite-differences (rbf-fd) on multiple gpus. Journal of Computational Physics, 231(21):7133 – 7151, 2012.