Appendices

Appendix A

A MATLAB code is presented for the N - th order PD differentiation of a function with M variables, i.e., $f = f(x_1, x_2, x_3, x_4, \ldots, x_M)$. At the end of the code, a sample input is described below for up to the second order derivatives of a function defined as $f = x_1^2 + x_2^2 + x_3^2 + x_4^2$ with $0 \leq x_1$,x_2,x_3,$x_4 \leq 1$.

E. Madenci et al., *Peridynamic Differential Operator for Numerical Analysis*,
https://doi.org/10.1007/978-3-030-02647-9

```
clc;
clear;
close all;
echo off;
global icount;
global p;
global ndiv;
global coord;
global dx;
global M;
global N;
%
% Input parameters
%
% M is currently limited to 10.
% If M>10 increase the size of arrays num[] and num1[] everywhere.
M = str2num(input('Specify the number of dimensions= ','s'));
N = str2num(input('Specify the maximum polynomial order in TSE= ','s'));
%
fprintf ('Specify the domain length for each dimension\n')
length = zeros(M,1);
for i=1:M
   fprintf('x%d ',i);
   length(i,1) = str2num(input('= ' ,'s'));
end
%
fprintf ('Specify the number of intervals for each dimension\n')
ndiv = zeros(M,1);
for i=1:M
   fprintf('ndivx%d ',i);
   ndiv(i,1) = str2num(input('= ' ,'s'));
end
%
fprintf ('Specify the order of derivatives for each dimension\n')
porder = zeros(M,1);
for i=1:M
   fprintf('p%d ',i);
```

```
    porder(i,1) = str2num(input(   '= ' ,'s'));
end
%
% Compute total number of PD points
totnode=  prod(ndiv);
% Initialize coordinate array
coord=zeros(totnode,M);
% Compute the number of TS terms based on N
icount=0;
loop1(1,N);
nsize=icount;        % nsize = size of PDDO (number of TS terms)
%
p=zeros(nsize,M);     % 2d array storing polynomial orders     of TS terms
num=zeros(10,1);
icount=0;
loop2(1,N,num);       % Assign polynomial orders in each TS term.
bb = zeros(nsize,1);
for ii=1:nsize
    bb(ii)=1;
    for mm=1:M
        bb(ii) = bb(ii)*factorial(p(ii,mm));
    end
end
%
dmag = 0;
dEntity = 1;
dx=zer os(M,1);       % Initialize interval size array
for ii=1:M
    dx(ii) = length(ii)/ndiv(ii);     % Compute interval size
    delta(ii) = dx(ii)*(N+1);         % Compute horizon size in each dimension
    dmag = dmag + delta(ii)*delta(ii);    % Compute max. size of h   orizon
    dEntity = dEntity*dx(ii);         % Compute jacobian of domain integration
end
dmag = sqrt(dmag);
num=zeros(10,1);
icount=0;

% Generate peridynamic grid and store the coordinates to coord array.
loop3(1,num);
%
% fvec (input): Function values at p   eridynamic points
fvec = zeros(totnode,1);
for k=1:totnode
    % Specify function
    % f(x1,x2,x3,x4) = x1^2 + x2^2 + x3^2 + x4^2      -- Example
    fvec(k) = coord(k,1)^2 + coord(k,2)^2 + coord(k,3)^2 + coord(k,4)^2;
```

```
end
%
% Initialize family array of a material point.
% Increase the size of this array if family members exceed 10000.
nodefam = zeros(10000,1);
% Array storing abs(xsi(m),m=1,M)
idist=zeros(M,1);
nmax = 0;
% Array storing xsi(1)^p1*xsi(2)^p2*...*xsi(M)^pM for each TS term
pvec = zeros(nmax,1) ;
% Array storing weight for each TS term
weight = zeros(nmax,1);
fileID = fopen('PDDO.out','w');
% Array storing output of PDDO
dfvec=zeros(totnode,1);
for k=1:totnode
   if int32(k)/int32(1000)*int32(1000) == int32(k)
      fprintf("k = %d\n",k)
   end
   %
   % Generate family of material point k
   numfam =1;
   nodefam(1) = k;
   for j = 1:totnode
      if j == k
         continue;
      end
      for mm=1:M
         idist(mm) = abs(coord(j,mm) - coord(k,mm));
      end
      inside = true;
      for mm=1:M
         if idist(mm) > delta(mm)
            inside=false;
            break;
         end
      end
      if inside
         numfam = numfam + 1;
         nodefam(numfam) = j;
      end
   end
   % fprintf("k = %d , numfam = %d\n", k, numfam)
   % End of generation of family of material point k
   %
   if(numfam>nmax)
```

```
    nmax = numfam;
    fprintf("numfam = %d\n",numfam);
end
%
% Compute shape matrix Amat
Amat = zeros(nsize,nsize);
bvec = zeros(nsize,1);
for kk=1:numfam
    j = nodefam(kk);
    xsimag = 0;
    for mm=1:M
        xsi(mm) = coord(j,mm) - coord(k,mm);
        xsimag = xsimag + xsi(mm)*xsi(mm);
    end
    xsimag = sqrt(xsimag);
    for ii=1:nsize
        pvec(ii) = 1.0;
        for mm=1:M
            pvec(ii) = pvec(ii)*xsi(mm)^p(ii,mm);
        end
        weight(ii) = exp(-4*(xsimag/dmag)^2);
        %weight[ii] = 1.0;
    end
    for ii=1:nsize
        for jj=1:nsize
            Amat(ii,jj) = Amat(ii,jj) + ...
                weight(ii)*pvec(ii)*pvec(jj)*dEntity;
        end
    end
end
% End of computation of shape matrix Amat
%
% Invert shape matrix and store the inversion to AmatInv
AmatInv = inv(Amat);
%
% Compute rhs vector for computing PD function for requested porder
for ii=1:nsize
    imatch = true;
    for mm=1:M
        if p(ii,mm)~= porder(mm)
            imatch = false;
            break;
        end
    end
    if imatch
        bvec(ii) = bb(ii);
```

```
            break;
        end
    end
    %
    % Compute the coefficients for compution PD function associated with
    % porder
    avec = AmatInv*bvec;
    %
    %
    % Compute derivative based on specified porder
    dfval = 0;
    for kk=1:numfam
        j = nodefam(kk);
        ff = fvec(j);
        xsimag = 0;
        for mm=1:M
            xsi(mm) = coord(j,mm) - coord(k,mm);
            xsimag = xsimag + xsi(mm)*xsi(mm);
        end
        xsimag = sqrt(xsimag);
        %
        % Compute PD function
        gfun = 0;
        for ii=1:nsize
            pvec(ii) = 1;
            for mm=1:M
                pvec(ii) = pvec(ii)*xsi(mm)^p(ii,mm);
            end
            weight(ii) = exp(-4*(xsimag/dmag)^2);
            gfun = gfun + avec(ii)*weight(ii)*pvec(ii);
        end
        % Compute the derivative from PD integration
        dfval = dfval + fvec(j)*gfun*dEntity;
    end
    dfvec(k) = dfval;
    for mm=1:M
        fprintf(fileID,"%f , ",coord(k,mm));
    end
    fprintf(fileID,"%f , %f\n",fvec(k),dfvec(k));
end
%
% Function to compute factorial of an integer number
function f = factorial1(n)
if(n<=1)
    f=1;
else
```

```
    f = n*factorial1(n-1);
end
end
%
% Function to count the total number of TS terms
function loop1(m, n)
global M;
global icount;
if(m>0) && (m<=M)
   for i=0:n
      loop1(m+1,n-i);
      if m==M
         icount=icount+1;
      end
   end
end
end
%
% Function to identify and assign the powers of TS terms
function loop2(m, n, num)
global icount;
global p;
global M;
num1=zeros(10,1);
for ii=1:10
   num1(ii)=num(ii);
end
if(m>0) && (m<=M)
   for i=0:n
      num1(m) = i;
      loop2(m+1,n-i,num1);
      if m==M
         icount=icount+1;
         for m=1:M
            p(icount,m) = num1(m);
         end
      end
   end
end
end
%
% Function to generate the grid of the multidimensional domain
function loop3(m, num)
global icount;
global ndiv;
global coord;
```

```
global M;
global dx;
num1 = zeros(10,1);
for ii=1:10
   num1(ii)=num(ii);
end
if(m>0 && m<=M)
   for i=1:ndiv(m)
      num1(m)=i;
      loop3(m+1,num1);
      if(m==M)
         icount=icount+1;
         for ii=1:M
            coord(icount,ii) = dx(ii)/2.0+(num1(ii)-1)*dx(ii);
         end
      end
   end
end

end
```

INPUT
Specify the number of dimensions= 4
Specify the maximum polynomial order in TSE= 2
Specify the domain length for each dimension
x1 = 1
x2 = 1
x3 = 1
x4 = 1
Specify the number of intervals for each dimension
ndivx1 = 10
ndivx2 = 10
ndivx3 = 10
ndivx4 = 10
Specify the order of derivatives for each dimension
p1 = 0
p2 = 1
p3 = 1
p4 = 2

Appendix B

The BiConjugate Gradient Stabilized (BICGSTAB) algorithm is a well-known iterative technique for solving sparse non-symmetric linear systems. It was developed by Van der Vorst (1992) based on the conjugate gradient (CG) and BiConjugate (BCG) methods.

The discrete form of the PD governing equations can be cast as a linear system of algebraic equations as

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{B.1}$$

in which $\mathbf{A} = \mathbf{H}$, $\mathbf{x} = \mathbf{V}$, and $\mathbf{b} = \mathbf{R}$. The algorithm of BICGSTAB is given by

1. Initialize unknown vector: $\mathbf{x}_0 = \mathbf{0}$.
2. Initial residual vector: $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$.
3. Choose an arbitrary vector $\tilde{\mathbf{r}}_0 = \mathbf{r}_0$.
4. $\rho_0 = \alpha = \omega_0 = 1$.
5. Initialize search directions:$\mathbf{v}_0 = \mathbf{p}_0 = \mathbf{0}$.
   Iteration : $k = 0,1,2,\ldots$
6. $$\rho_k = \tilde{\mathbf{r}}_0 \cdot \mathbf{r}_{k-1}.$$
7. $$\beta = (\rho_k/\rho_{k-1})(\alpha/\omega_{k-1}).$$
8. Update search direction: $\mathbf{p}_k = \mathbf{r}_{k-1} + \beta(\mathbf{p}_{k-1} - \omega_{k-1}\mathbf{v}_{k-1})$.
9. Update search direction: $\mathbf{v}_k = \mathbf{A}\mathbf{p}_k$.
10. $$\alpha = \rho_k/\left(\tilde{\mathbf{r}}_0 \cdot \mathbf{v}_k\right).$$
11. $$\mathbf{s} = \mathbf{r}_{k-1} - \alpha\mathbf{v}_k.$$
12. $$\mathbf{t} = \mathbf{A}\mathbf{s}.$$
13. $$\omega_k = (\mathbf{t} \cdot \mathbf{s})/(\mathbf{t} \cdot \mathbf{t}).$$
14. Update the solution: $\mathbf{x}_k = \mathbf{x}_{k-1} + \alpha\mathbf{p}_k + \omega_k\mathbf{s}$.
15. Residual vector: $\mathbf{r}_k = \mathbf{s} - \omega_k\mathbf{t}$.
16. If $(\|\mathbf{r}_k\|/\|\mathbf{b}\|) < \varepsilon$, stop the iteration. If not, $k = k+1$and go back to Step 6

where $\varepsilon = 10^{-5}$ is the tolerance.

Appendix C

The adaptive time stepping method introduced by Zohdi (2013) adjusts the time step size by ensuring the stability of the numerical time integrations. The methodology is explained by considering the following transient equations:

$$\mathbf{M}\ddot{\mathbf{V}}(t) = \boldsymbol{\Psi}(t) \tag{C.1a}$$

and

$$\mathbf{M}\dot{\mathbf{V}}(t) = \boldsymbol{\Psi}(t) \tag{C.1b}$$

in which $\mathbf{M}$ is the coefficient matrix, $\mathbf{V}$is the unknown field vector, and $\boldsymbol{\Psi}$ is the known vector. According to adaptive time stepping method, these equations can be discretized in time as

$$\mathbf{V}^{t+\Delta t} = \mathbf{V}^{t} + \dot{\mathbf{V}}^{t}\Delta t + \phi\Delta t^2\mathbf{M}^{-1}\left(\phi\boldsymbol{\Psi}^{t+\Delta t} + (1-\phi)\boldsymbol{\Psi}^{t}\right) \tag{C.2a}$$

and

$$\mathbf{V}^{t+\Delta t} = \mathbf{V}^{t} + \phi\Delta t\mathbf{M}^{-1}\left(\phi\boldsymbol{\Psi}^{t+\Delta t} + (1-\phi)\boldsymbol{\Psi}^{t}\right) \tag{C.2b}$$

The parameter, Δt, is the time step, and $\mathbf{V}^t$ is the known field vector at time t. Its time derivatives, $\ddot{\mathbf{V}}^t$ and $\dot{\mathbf{V}}^t$, are approximated as

$$\ddot{\mathbf{V}}^{t} = \frac{\dot{\mathbf{V}}^{t} - \dot{\mathbf{V}}^{t-\Delta t}}{\Delta t} \tag{C.3a}$$

and

$$\dot{\mathbf{V}}^{t} = \frac{\mathbf{V}^{t} - \mathbf{V}^{t-\Delta t}}{\Delta t} \tag{C.3b}$$

Also, the parameter ϕ varies between 0 and 1. Equation (C.2) reduces to an implicit form when ϕ=1 and an explicit form when ϕ=0. Any other value of ϕ yields its partially explicit and implicit form. In this book, ϕ is specified as 1 (implicit form); thus, Eq. (C.2) is solved iteratively until achieving the optimum step size. The corresponding solutions to Eqs. (C.2a) and (C.2b) can be written as

$$\mathbf{V}^{t+\Delta t_L} = \mathbf{V}^{t} + \dot{\mathbf{V}}^{t}\Delta t_L + \Delta t_L^2\mathbf{M}^{-1}\boldsymbol{\Psi}^{t+\Delta t_{L-1}} \tag{C.3b}$$

$$\mathbf{V}^{t+\Delta t_L} = \mathbf{V}^t + \dot{\mathbf{V}}^t \Delta t_L + \Delta t_L^2 \mathbf{M}^{-1} \boldsymbol{\Psi}^{t+\Delta t_{L-1}} \tag{C.4a}$$

and

$$\mathbf{V}^{t+\Delta t_L} = \mathbf{V}^t + \Delta t_L \mathbf{M}^{-1} \boldsymbol{\Psi}^{t+\Delta t_{L-1}} \tag{C.4b}$$

where L is the iteration number, and Δt_L (with $\Delta t_0 = 0$) denotes the step size after L-th iteration at time t. The optimum step size Δt_L which leads to stable solution is achieved by the following algorithm:

1. Set $L = 1$ at time t,
2. Compute the new field vector from Eq. (C.4),
3. Measure normalized error quantities as

   (a)
   $$\varpi^L = \frac{\left\| \mathbf{V}^{t+\Delta t_L} - \mathbf{V}^{t+\Delta t_{L-1}} \right\|}{\left\| \mathbf{V}^{t+\Delta t_L} \right\|}$$

   (b)
   $$Z^L = \frac{\varpi^L}{\varepsilon}$$

   (c)
   $$\Phi^L = \left(\frac{\left(\frac{TOL}{\varpi^0}\right)^{\frac{1}{pL_d}}}{\left(\frac{\varpi^L}{\varpi^0}\right)^{\frac{1}{pL}}} \right)$$

4. If the tolerance is satisfied, i.e., [($Z^L \leq 1$) and $L < L_d$]
   (a) Construct a new time step: $\Delta t_L = \Phi^L \Delta t_{L-1}$.
   (b) Select the minimum time step: $\Delta t_{\text{new}} = \min(\Delta t_{\text{lim}}, \Delta t_{L+1})$.
   (c) Increment time: $t = t + \Delta t_{\text{new}}$, and go to (1).
5. If the tolerance is not satisfied, i.e., [($Z^L > 1$) and $L \leq L_d$]
   (a) Construct a new time step: $\Delta t_L = \Phi^L \Delta t_{L-1}$.
   (b) Select the minimum time step: $\Delta t_{\text{new}} = \min(\Delta t_{\text{lim}}, \Delta t_{L+1})$.
   (c) Restart from time t and go to (1).

where L_d is the number of desired iterations, generally chosen to be between 5 and 10 iterations, Δt_{lim} is the maximum step size allowed which is taken as 10^{-3}, and ε is the tolerance and specified as 10^{-5}. Furthermore, due to the quadratic dependency on Δt_L, the parameter p is considered as 2.

References

Agwai, A (2011) A peridynamic approach for coupled fields. Ph.D. dissertation, Department of Aerospace and Mechanical Engineering, University of Arizona

Alexa M, Behr J, Cohen-Or D, Fleishman S, Levin D, Silva CT (2003) Computing and rendering point set surfaces. IEEE Transactions on visualization and computer graphics 9: 3–15

Alexiades V, Solomon AD (1993) Mathematical modeling of melting and freezing processes. Hemisphere Publishing Corporation, Washington, DC

Aluru NR (2000) A point collocation method based on reproducing kernel approximations. Int J for Numer Meth in Eng 47:1083–1121

Anderson TL (2005) Fracture Mechanics: Fundamentals and Applications. CRC Press, Boca Raton, FL

Apostol TM (1967) Calculus: One-variable calculus, with an introduction to linear algebra. John Wiley, New York

Armero F, Simo JC (1992) A new unconditionally stable fractional step method for non-linear coupled thermomechanical problems. Int J Numer Meth Engng 35:737–766

Ascher U, Mattheij R, Russell R. (1995) Numerical Solution of Boundary Value Problems for Ordinary Differential Equations, SIAM

Beléndez A, Hernández A, Márquez A, Beléndez T, Neipp C (2006) Analytical approximations for the period of a nonlinear pendulum. Eur J Phys 27:539

Belytschko T, Lu YY, Gu L (1994) Element-free Galerkin methods. International journal for numerical methods in engineering 37: 229–56

Belytschko T, Krongauz Y, Organ D, Fleming M, Krysl P (1996) Meshless methods: an overview and recent developments. Computer methods in applied mechanics and engineering 139: 3–47

Bender CM, Orszag SA (1999) Advanced Mathematical Methods for Scientists and Engineers I: Asymptotic Methods and Perturbation Theory. Springer, New York

Bessa M, Foster JT, Belytschko T, Liu WK (2014) A meshfree unification: Reproducing kernel peridynamics, Comp Mech 53:1251–1264

Bevington PR, Robinson DK (2003) Data Reduction and Error Analysis for the Physical Sciences. McGraw-Hill, New York

Biles DC, Robinson MP, Spraker JS. (2002) A generalization of the Lane–Emden equation. J Math Anal App 273:654–666

Bobaru F, Yang M, Alves LF, Silling SA, Askari E, Xu J (2009) Convergence, adaptive refinement, and scaling in 1D peridynamics. Int J Numer Meth Eng 77:852–877.

Bocher M (1974) Integral Equations. Cambridge University Press, London,

Chen JS, Wu CT, Belytschko T. (2000) Regularization of material instabilities by meshfree approximations with intrinsic length scales. Int J Numer Meth Eng 47:1303–1322

E. Madenci et al., *Peridynamic Differential Operator for Numerical Analysis*,
https://doi.org/10.1007/978-3-030-02647-9

Chen JS, Zhang X, Belytschko T (2004) An implicit gradient model by a reproducing kernel strain regularization in strain localization problems. Comput Methods in Appl Mech Eng 193:2827–2844.

Chi SW, Chen JS, Hu HY, Yang JP (2013) A gradient reproducing kernel collocation method for boundary value problems. Int J Numer Meth Eng 93:1381–402

Ding H, Shu C, Yeo KS, Xu D (2004a) Development of least-square-based two-dimensional finite-difference schemes and their application to simulate natural convection in a cavity. Computers & fluids 33: 137–154

Ding H, Shu C, Yeo KS, Xu D (2004b) Simulation of incompressible viscous flows past a circular cylinder by hybrid FD scheme and meshless least square-based finite difference method. Computer Methods in Applied Mechanics and Engineering 193: 727–744

Ferziger JH, Peric M (2002) Computational methods for fluid dynamics. Springer, Berlin

Gavete L, Ureña F, Benito JJ, García A, Ureña M, Salete E (2017) Solving second order non-linear elliptic partial differential equations using generalized finite difference method. Journal of Computational and Applied Mathematics 318:378–387

Geng F (2012) Iterative reproducing kernel method for a beam equation with third-order nonlinear boundary conditions. Math Sci 6:1–4

Gingold RA, Monaghan JJ (1977) Smoothed particle hydrodynamics: theory and application to non-spherical stars. Mon Not Roy Astron Soc 181:375–389

Haberman R (1998) Elementary applied partial differential equations: With fourier series and boundary value problems, Prentice Hall, Upper Saddle River, NJ

Haji-Sheikh A (2013) Exact Analytical Conduction Toolbox.

He JH (1999) Variational iteration method–a kind of non-linear analytical technique: some examples, International Journal of Non-Linear Mechanics. 34:699–708.

He JH (2000) Variational iteration method for autonomous ordinary differential systems. Applied Mathematics and Computation 114:115–123.

He JH (2003) Homotopy perturbation method: a new nonlinear analytical technique, Applied Mathematics and Computation 135:73–79.

He JH (2006) Some asymptotic methods for strongly nonlinear equations, International Journal of Modern Physics B, 20:1141–1199.

He JH (2007) Variational iteration method–some recent results and new interpretations, J. Comput. Appl. Math. 207:3–17.

Huan YJ, Liu HK (2013) A new modification of the variational iteration method for van der Pol equations, 37:8118–8130.

Huang YQ, Li QS (2004) Bending and buckling analysis of antisymmetric laminates using the moving least square differential quadrature method. Computer methods in applied mechanics and engineering 193: 3471–3492

Hulbert GM, Hughes TJR (1990) Space-time finite element methods for second-order hyperbolic equations. Comput Methods Appl Mech Eng 84:327–348

Jerri AJ (1999) Introduction to Integral Equations with Applications. Wiley, New York

Lancaster P, Salkauskas K (1981) Surfaces generated by moving least squares methods. Mathematics of computation 37:141–58

Li S, Liu WK (1999) Reproducing kernel hierarchical partition of unity, Part II—applications. Int J Numer Meth Engng 45:289–317

Li H, Huang Y, Li XF (2008) Approximate solution of Abel integral equation, Computers & Mathematics with Applications, 56:1748–1757.

Li C, He-Ping M, Yu-Min C (2013) Combining the complex variable reproducing kernel particle method and the finite element method for solving transient heat conduction problems. Chinese Phys B 22(5):050202

Liew KM, Huang YQ, Reddy JN (2002) A hybrid moving least squares and differential quadrature (MLSDQ) meshfree method. International Journal of Computational Engineering Science 3: 1–2

Liszka T (1984) An interpolation method for an irregular net of nodes. International Journal for Numerical Methods in Engineering 20: 1599–1612
Liszka T, Orkisz J (1980) The Finite Difference Method at Arbitrary Irregular Grids and Its Application in Applied Mechanics. Computers and Structures 11: 83–95.
Liu CS, Atluri S.N (2009), A fictitious time integration method for the numerical solution of the Fredholm integral equation and for numerical differentiation of noisy data, and its relation to the filter theory, CMES: Computer Modeling in Engineering 41:243–261.
Liu WK, Jun S, Zhang YF (1995) Reproducing kernel particle methods. Int J Numer Meth Fluids 20:1081–1106
Liu WK, Li S, Belytschko T (1997) Moving least-square reproducing kernel methods (I) methodology and convergence. Computer methods in applied mechanics and engineering 143: 113–54
Liu Y, Zhang X, Lu MW (2005) Meshless least-squares method for solving the steady-state heat conduction equation. Tsinghua Science & Technology 10: 61–66
Lucy LB (1977) A numerical approach to the testing of the fission hypothesis. Astronomical J 82:1013–1024
Mackowski DW (2016) Conduction Heat Transfer Notes for MECH 7210, Mech. Engr. Dept., Auburn University, http://www.eng.auburn.edu/~dmckwski/mech7210/condbook.pdf
Madenci E, Barut A, Futch M (2016) Peridynamic differential operator and its applications. Comput Methods in Appl Mech Eng 304:408–451
Madenci E, Dorduncu M, Barut A, Futch M (2017) Numerical solution of linear and nonlinear partial differential equations using the peridynamic differential operator. Numer Methods Partial Differ Equ 33:1726–1753
Madenci E, Dorduncu M, Gu X., (2018) Peridynamic least squares minimization. Computer Methods in Applied Mechanics and Engineering (under review)
Mukherjee YX, Mukherjee S (1997) On boundary conditions in the element-free Galerkin method. Comput Mech 19:264–270
Nayroles B, Touzot G, Villon P (1992) Generalizing the finite element method: diffuse approximation and diffuse elements. Computational mechanics 10: 307–18
Nealen A (2004) An as-short-as-possible introduction to the least squares, weighted least squares and moving least squares methods for scattered data approximation and interpolation. URL: http://www.nealen.com/projects
Nguyen H, Reynen J (1984) A space-time least-square finite element scheme for advection-diffusion equations. Computer Methods in Applied Mechanics and Engineering 42: 331–342
Nickell RE, Sackman JL (1968) Approximate Solutions in Linear, Coupled Thermoelasticity. J Appl Mech 35:255–266
Onate E, Idelsohn S, Zienkiewicz OC, Taylor RL (1996) A finite point method in computational mechanics. Applications to convective transport and fluid flow. International journal for numerical methods in engineering 39: 3839–3866.
Pan XF, Zhang X, Lu MW (2005) Meshless Galerkin least-squares method. Computational Mechanics 35: 182–189
Pérez-Aparicio JL, Taylor RL, Gavela D (2007) Finite element analysis of nonlinear fully coupled thermoelectric materials. Comput Mech 40:35–45
Randall JL (2005) Finite difference methods for differential equations. A Math 585–586 Lecture Notes, University of Washington
Roache PJ (1994) Perspective: A Method for uniform reporting of grid refinement studies. J Fluids Eng 116:405–413
Salehi R, Dehghan M (2013) A generalized moving least square reproducing kernel method. Journal of Computational and Applied Mathematics 249: 120–32
Silling SA (2000) Reformulation of elasticity theory for discontinuities and long-range forces. Journal of the Mechanics and Physics of Solids 48:175–209.
Silling SA Epton M, Weckner O, Xu J, Askari E. (2007) Peridynamic States and Constitutive Modeling. Journal of Elasticity 88:151–184.

Spijker MN. (1996) Stiffness in numerical initial-value problems. J Comp App Math 72:393–406
Srivastava VK, Tamsir M, Awasthi MK, Singh S (2014) One-dimensional coupled Burgers' equation and its numerical solution by an implicit logarithmic finite-difference method. AIP ADVANCES 4: ADV037119–10
Strutz T (2010) Data fitting and uncertainty: A practical introduction to weighted least squares and beyond. Vieweg and Teubner
Tandon GP, Pochiraju KV, Schoeppner GA (2008) Thermo-oxidative behavior of high-temperature PMR-15 resin and composites. Materials Science and Engineering: A 498:150–161.
Van Der Vorst HA (1992) Bi-CGStab: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. SIAM J Sci Stat Comput 13(2):631–44.
Ventsel E, Krauthammer T (2001) Thin Plates and Shells: Theory: Analysis, and Applications. CRC Press, New York
Voller V, Cross M (1981) Accurate solutions of moving boundary problems using the enthalpy method. International Journal of Heat and Mass Transfer 24(3):545–556.
Walker JDA (1978) The boundary layer due to a rectilinear vortex. Proc. Roy. Soc. Lond. A 359:167–188
Walker JDA, Weigand GG (1979) An accurate method for two-point boundary value problems. Int J Numer Meth Eng 14:1335–1346
Wazwaz AM (1997) A First Course in Integral Equations, World Scientific, Singapore.
Wazwaz AM (2011) Linear and nonlinear integral equations: methods and applications. Springer, New York
Wazwaz AM (2014) The variational iteration method for solving linear and nonlinear ODEs and scientific models with variable coefficients Cent. Eur. J. Eng., 4:64–71.
Wertheim GK, Dicenzo SB (1985) Least-squares analysis of photoemission data. Journal of electron spectroscopy and related phenomena 37: 57–67
Zhu T, Zhang J, Atluri SN (1998) A local boundary integral equation (LBIE) method in computational mechanics, and a meshless discretization approach. Computational Mechanics 21:223–235
Zohdi TI (2013) Numerical simulation of the impact and deposition of charged particulate droplets. Journal of Computational Physics 233:509–526.
Zuo P, Zhao Y (2014) A phase field model coupling lithium diffusion and stress evolution with crack propagation and application in lithium ion batteries. Phys Chem Chem Phys 17:287–297.

Index

A
Adaptive time-stepping algorithm, 120, 129, 165, 238, 272
Angular displacement, 92
Avogadro's constants, 176

B
Bar, 118, 123, 127, 132, 135, 163, 169
Beam, 105, 113
  buckling, 110
  buckling load, 110
  critical buckling load, 110
  initial imperfection, 113
Bessel function, 136
BiConjugate Gradient Stabilized (BICGSTAB) method, 223
Boltzmann constant, 176
Boundary conditions
  clamped, 110
  concentrated load, 114, 149
  Dirichlet- (essential), 99, 136, 141, 209
  distributed axial load, 113
  distributed transverse load, 113
  flux, 118, 138, 224
  insulated, 139
  Neumann- (natural), 99, 141, 209
  nonlinear, 105
  quasi-static loading, 238
  Robbin-type, 99
  simply supported, 110, 145
  temperature, 138
  time-dependent, 120
  traction free condition, 149, 179
  traction load, 239
  uniform pressure, 145
Boundary layer equation, 103
Boundary layer flow, 99
Boundary value problems (BVP), 99
Burger's equation, 159

C
Cauchy's stress tensor, 224
Central difference method, 134
Classical continuum mechanics (CCM), 224
  equation of motion, 224
Convergence, 46
  order of accuracy, 47
  rate of, 47
Coupled field equation, 159
Crack, 138, 149
  insulated, 138
  length, 138, 149, 178
  opening, 153, 241
  pre-existing, 152, 241
  surface, 139, 152, 153
  tip, 157, 241

D
Diffuse finite element method (DFEM), 247
Diffusivity
  varying, 123
Dirac delta function, 125
Discretization, 39
Displacement, 132, 149, 164, 177, 238
  gradient tensor, 224
Dissimilar materials, 124
Divergence theorem, 212

E. Madenci et al., *Peridynamic Differential Operator for Numerical Analysis*,
https://doi.org/10.1007/978-3-030-02647-9

E
Eigenvalue, 111
Eigenvector, 112
Electrical potential, 169
Electric conductivity, 169
Element free Galerkin method (EFGM), 247
Enthalpy, 128, 129
Error function, 46
  actual, 49
  estimator, 47
  relative, 49

F
Finite difference method (FDM), 1, 2, 247
Finite element method (FEM), 1, 2
Finite point method (FP), 247
Finite volume method (FVM), 2
Flexural rigidity, 145
Fluid flow, 159
Flux, 123, 211
Frequency, 92

G
Gas dynamics, 159
Gaussian distribution, 49
Generalized finite difference method (GFDM), 2, 3, 248
Generalized moving least squares reproducing kernel method (GMLSRKM), 248
Ghost particles, 248
Global error measure, 47
Gradient reproducing kernel (G-RK), 5
Gravity, 92

H
Heat conduction, 138, 141
  heat source, 138
Heat diffusion, 118, 120, 123
Heaviside step function, 124, 129, 132
Helmholtz equation, 135
He's polynomials, 205, 206
Homotopy method, 205
Horizon, 3, 7, 8, 45, 47, 49, 210
  characteristic length, 210
  family members, 6, 8, 211
  nonsymmetric family, 7, 14–18, 24–27, 31–33, 35–36, 39, 120
  symmetric family, 7, 19–23, 27–31, 33–37, 39

I
Image compression, 72
Image recovery, 71
Implicit time integration, 119, 122
Initial value problem (IVP), 91
Integral equation (IE)
  Abel's, 1, 188–190
  Fredholm, 197, 199
  Volterra, 1, 199
Integro-differential equation (IDE), 187
  Fredholm, 187, 194, 195, 203
  Volterra, 187, 194
Interface, 123, 126, 130
  moving, 128
Internal force vector, 224
Internal length, 7, 19
Interpolation, 57
  Lagrangian function, 57
  one-dimensional, 59
  polynomials, 57
  spline, 57
  two-dimensional, 65

J
Jacobi elliptic function, 92

L
Lagrange multipliers, 42, 44, 194, 198, 200, 202, 206, 249
Lane-Emden equation, 93
Laplace transformation, 132
Latent heat, 128
Least squares finite element method (LSFEM), 247
Least squares minimization (LSM), 3, 247
Leibniz's rule, 188
Lithium diffusion (lithiation), 175

M
Mass matrix, 238
Mesh-free least squares finite difference method (MLSFDM), 238
Meshless Galerkin least squares method (MGLS), 247
Meshless quadrature technique, 39
Meshless weighted least squares method (MWLS), 247
Molecular mobility, 176
Moment matrix, 6

Moving least squares differential quadrature method (MLSDQM), 248
Moving least squares minimization (MLSM), 247

N
Natural convection, 107
Newton-Raphson method, 44, 91, 162, 237
  incremental residual vector, 43
  initial guess, 91
  residual vector, 42
  tolerance, 91
Noise removal, 81
Noisy data, 57
Nonlinear acoustic, 159
Numerical integration, 46
  integration error, 46, 54

O
Ordinary differential equation (ODE), 4, 39, 42, 91
  coupled differential equation, 96, 107, 113
Orthogonality property, 8–10
Oxygen concentration, 173

P
Partial differential equation (PDE), 4, 39, 117
  elliptic equation, 117
  hyperbolic equation, 117
  parabolic equation, 117
Partial polar volume, 176
Pascal's pyramid, 252
Pendulum, 91
Peridynamic differential operator (PDDO), 3, 5, 7
  peridynamic functions, 8, 10, 13, 24, 30, 45, 59
Peridynamics (PD), 3, 6
  differentiation, 5, 42, 45, 48
  material parameters, 7
Phase change, 126
Pixel, 71, 74
Plane stress, 148, 176, 231
Plate, 138, 141, 145, 148, 175, 223, 239
Poisson's equation, 211
Potential vortex, 99

R
Regression, 57, 75, 86
  one-dimensional, 77
  two-dimensional, 78
Reproducing kernel particle method (RKPM), 2, 3, 248
Reproducing kernel (RK), 5
Reproducing kernel strain regularization (RKSR), 6

S
Seebeck coefficient, 169
Shape matrix, 9, 12, 14, 24, 259
Singular perturbation, 100
Smoothed particle hydrodynamics (SPH), 2, 3
Smoothing, 57, 80
  recursive, 83
Spatial derivative, 39
Specific heat, 129
Status parameter, 139, 214, 228
Stefan problem, 126
  Stefan (interface) condition, 128
Stiff equation, 95, 204
Stress field, 157
  hydrostatic, 175
  remote stress, 241

T
Taylor series expansion (TSE), 8, 10
  number of terms, 45, 48
  remainder, 10, 45, 47, 249
  truncation error, 45
Temperature, 118, 123, 127, 138, 141, 164, 169, 224
  absolute, 176
  melting, 128
  reference, 164
Temporal derivative, 39
Tension
  tensile load, 149
  uniform, 239
Thermal conductivity, 128, 169
Thermal coupling coefficient, 164
Thermal oxidation, 173
  aging temperature, 173
  chemical reaction rate, 173
  diffusion coefficient, 173
  oxidation state variable, 173

Thermoelastic wave, 163
Thermoelectricity, 169
Thermoelectric materials, 169
Time step size, 238, 272
Traffic flow, 159

V
Van der Pol equation, 204
Variational form, 213
Variational iteration method (VIM), 4, 193, 198, 201
  correction functional, 193, 194, 196, 198, 200, 202
Variational theory, 194
Velocity, 238
Virtual work, 227, 234

W
Wave, 132
  dispersion, 133
  elastic, 168
  elastic wave speed, 164
  number, 135, 137
  propagation, 132
  thermal, 168
Weighted least squares minimization (WLSM), 247
Weight function, 6, 9, 11, 49, 59, 249