

```

(*****)
(* *)
(* Generalized Collocation Methods: Solutions to Nonlinear Problems *)
(* Bellomo, N., Lods, B., Revelli, R., Ridolfi, L. *)
(* A Birkhäuser book *)
(* ISBN: 978-0-8176-4525-0 *)
(* *)
(* Program SlabTwoDim *)
(* *)
(*****)

$TextStyle = {FontFamily -> "Times", FontSize -> 12};
Off[General::"spell", General::"spell1"]

SlabTwoDim[InitBoundDataX_, InitBoundDataY_,
  NodesX_, NodesY_, TypeX_, TypeY_] := Module[{ },

  (** INITIAL CONDITION **)

   $\varphi[x_, y_] := \text{InitBoundDataX}[2];$ 

   $hhX = \frac{1}{\text{NodesX} - 1};$ 
   $hhY = \frac{1}{\text{NodesY} - 1};$ 

  Which[TypeX === Sinc, x1_i_ := (i - 1) * hhX,
    TypeX === Lagrange, x1_i_ :=  $-\frac{1}{2} \left( \cos\left[(i - 1) * \frac{\pi}{\text{NodesX} - 1}\right] - 1\right)$ ];
  Which[TypeY === Sinc, y1_i_ := (i - 1) * hhY, TypeY === Lagrange,
    y1_i_ :=  $-\frac{1}{2} \left( \cos\left[(i - 1) * \frac{\pi}{\text{NodesY} - 1}\right] - 1\right)$ ];

  If[TypeX === Lagrange, LagrX[j_, x_] :=  $\prod_{p=1}^{\text{NodesX}} \left( \text{If}[p \neq j, \frac{x - x1_p}{x1_j - x1_p}, 1] \right)$ ];

  If[TypeY === Lagrange, LagrY[j_, y_] :=  $\prod_{p=1}^{\text{NodesY}} \left( \text{If}[p \neq j, \frac{y - y1_p}{y1_j - y1_p}, 1] \right)$ ];

  If[TypeX === Sinc, SincX[j_, x_] := Which[ $0 \leq x \leq 1$  &&  $x \neq (j - 1) * hhX$ ,
     $\sin\left[\frac{\pi * (x - (j - 1) * hhX)}{hhX}\right] / \left(\frac{\pi * (x - (j - 1) * hhX)}{hhX}\right)$ ,  $x == (j - 1) * hhX$ , 1]];

  If[TypeY === Sinc, SincY[j_, y_] := Which[ $0 \leq y \leq 1$  &&  $y \neq (j - 1) * hhY$ ,
     $\sin\left[\frac{\pi * (y - (j - 1) * hhY)}{hhY}\right] / \left(\frac{\pi * (y - (j - 1) * hhY)}{hhY}\right)$ ,  $y == (j - 1) * hhY$ , 1]];

  If[TypeX === Lagrange && TypeY === Lagrange, InitValue[x_, y_] :=
     $\sum_{m=1}^{\text{NodesX}} \sum_{n=1}^{\text{NodesY}} ((\varphi[x, y] /. \{x \rightarrow x1_m, y \rightarrow y1_n\}) * \text{LagrX}[m, x] * \text{LagrY}[n, y])$ ];

  If[TypeX === Lagrange && TypeY === Sinc, InitValue[x_, y_] :=
     $\sum_{m=1}^{\text{NodesX}} \sum_{n=1}^{\text{NodesY}} ((\varphi[x, y] /. \{x \rightarrow x1_m, y \rightarrow y1_n\}) * \text{LagrX}[m, x] * \text{SincY}[n, y])$ ];

```

```

If[TypeX === Sinc && TypeY === Lagrange, InitValue[x_, y_] :=
  Sum[Sum[(phi[x, y] /. {x -> x1_m, y -> y1_n}) * SincX[m, x] * LagrY[n, y]],
    {m, 1, NodesX}, {n, 1, NodesY}];
If[TypeX === Sinc && TypeY === Sinc, InitValue[x_, y_] :=
  Sum[Sum[(phi[x, y] /. {x -> x1_m, y -> y1_n}) * SincX[m, x] * SincY[n, y]],
    {m, 1, NodesX}, {n, 1, NodesY}];

Print["Initial condition interpolation:"];

Plot3D[Evaluate[InitValue[x, y]],
  {x, 0, 1}, {y, 0, 1}, PlotRange -> All, AxesLabel -> {x, y, u}];

Print["Don't worry! I'm working"];

(*** Derivative matrices ***)
If[TypeX === Lagrange,
  FDMX[j_, i_] := Which[i == j, Sum[If[k == i, 0, 1/(x1_i - x1_k)],
    {k, 1, NodesX}],
    i != j, Sum[Product[If[p == i, 1, x1_i - x1_p], {p, 1, NodesX}],
    Product[If[p == j, 1, x1_j - x1_p], {p, 1, NodesX}]]];

FirstDerX = Table[FDMX[j, i], {j, 1, NodesX}, {i, 1, NodesX}];
SecondDerX =
  Table[If[i != j, 2 (FirstDerX[[j, i]] * FirstDerX[[i, i]] - 1/(x1_i - x1_j)), 0],
    {j, 1, NodesX}, {i, 1, NodesX}];
Do[SecondDerX[[i, i]] = - Sum[SecondDerX[[k, i]], {k, 1, NodesX}]];

If[TypeY === Lagrange,
  FDMY[j_, i_] := Which[i == j, Sum[If[k == i, 0, 1/(y1_i - y1_k)],
    {k, 1, NodesY}],
    i != j, Sum[Product[If[p == i, 1, y1_i - y1_p], {p, 1, NodesY}],
    Product[If[p == j, 1, y1_j - y1_p], {p, 1, NodesY}]]];

FirstDerY = Table[FDMY[j, i], {j, 1, NodesY}, {i, 1, NodesY}];
SecondDerY =
  Table[If[i != j, 2 (FirstDerY[[j, i]] * FirstDerY[[i, i]] - 1/(y1_i - y1_j)), 0],
    {j, 1, NodesY}, {i, 1, NodesY}];
Do[SecondDerY[[i, i]] = - Sum[SecondDerY[[k, i]], {k, 1, NodesY}]];

If[TypeX === Sinc,
  FDMX[j_, i_] := Which[i != j, (-1)^(i-j)/(hhX * (i - j)), i == j, 0];

```

```

SDMX[j_, i_] := Which[i != j,  $\frac{2 * (-1)^{i-j+1}}{hhX^2 * (i-j)^2}$ , i == j,  $-\frac{\pi^2}{3 * hhX^2}$ ];

FirstDerX = Table[FDMX[j, i], {j, 1, NodesX}, {i, 1, NodesX}];
SecondDerX = Table[SDMX[j, i], {j, 1, NodesX}, {i, 1, NodesX}];

If[TypeY == Sinc,

FDMY[j_, i_] := Which[i != j,  $\frac{(-1)^{i-j}}{hhY * (i-j)}$ , i == j, 0];

SDMY[j_, i_] := Which[i != j,  $\frac{2 * (-1)^{i-j+1}}{hhY^2 * (i-j)^2}$ , i == j,  $-\frac{\pi^2}{3 * hhY^2}$ ];

FirstDerY = Table[FDMY[j, i], {j, 1, NodesY}, {i, 1, NodesY}];
SecondDerY = Table[SDMY[j, i], {j, 1, NodesY}, {i, 1, NodesY}];

(***) Nodal Equations (***)

NdEq[i_, j_] := Ci,j'[t] == (1 - 2 Ci,j[t])
 $\left( \frac{1}{LL^2} \left( \sum_{k=1}^{NodesX} \text{FirstDerX}[[k, i]] * C_{k,j}[t] \right)^2 + \left( \sum_{k=1}^{NodesY} \text{FirstDerY}[[k, j]] * C_{i,k}[t] \right)^2 \right) +$ 
 $C_{i,j}[t] (1 - C_{i,j}[t]) \left( \frac{1}{LL^2} \left( \sum_{k=1}^{NodesX} \text{SecondDerX}[[k, i]] * C_{k,j}[t] \right) + \right.$ 
 $\left. \left( \sum_{k=1}^{NodesY} \text{SecondDerY}[[k, j]] * C_{i,k}[t] \right) \right) / . \{x \rightarrow x1_i, y \rightarrow y1_j\};$ 

EqOX = Flatten[Table[C1,j'[t] == D[InitBoundDataX[[1]], t] /. y → y1j, {j, 1, NodesY}]];
EqMX =
Flatten[Table[CNodesX,j'[t] == D[InitBoundDataX[[3]], t] /. y → y1j, {j, 1, NodesY}]];

EqOY =
Flatten[Table[Ci,1'[t] == D[InitBoundDataY[[1]], t] /. x → x1i, {i, 2, NodesX - 1}]];
EqMY = Flatten[Table[Ci,NodesY'[t] == D[InitBoundDataY[[3]], t] /. x → x1i,
{i, 2, NodesX - 1}]];

cEqOX = Flatten[Table[C1,j[0] == φ[x, y] /. {x → x11, y → y1j}, {j, 1, NodesY}]];
cEqMX = Flatten[Table[CNodesX,j[0] == φ[x, y] /. {x → x1NodesX, y → y1j}, {j, 1, NodesY}]];

cEqOY = Flatten[Table[Ci,1[0] == φ[x, y] /. {x → x1i, y → y11}, {i, 2, NodesX - 1}]];
cEqMY =
Flatten[Table[Ci,NodesY[0] == φ[x, y] /. {x → x1i, y → y1NodesY}, {i, 2, NodesX - 1}]];

EqsSys = Flatten[Table[{NdEq[i, j], Ci,j[0] == (φ[x, y] /. {x → x1i, y → y1j)},
{i, 2, NodesX - 1}, {j, 2, NodesY - 1}]];
NdEqsSys = Join[EqsSys, EqOX, EqMX, EqOY, EqMY, cEqOX, cEqMX, cEqOY, cEqMY];
UnKnown = Flatten[Join[Table[Ci,j, {i, 1, NodesX}, {j, 1, NodesY}]]];
NumSol = NDSolve[NdEqsSys, UnKnown, {t, 0, 1}, MaxSteps → 10000000] // Flatten;

(***) Solution (***)
aa = Partition[NumSol, NodesY];
If[TypeX == Lagrange && TypeY == Lagrange,

TotSol[x_, y_, t_] :=  $\sum_{m=1}^{NodesX} \sum_{n=1}^{NodesY} (aa[[m, n, 2]][t] * \text{LagrX}[m, x] * \text{LagrY}[n, y])$ ];

```

```

If[TypeX === Lagrange && TypeY === Sinc,
  TotSol[x_, y_, t_] :=  $\sum_{m=1}^{\text{NodesX}} \sum_{n=1}^{\text{NodesY}} (\text{aa}[[m, n, 2]][t] * \text{LagrX}[m, x] * \text{SincY}[n, y]);$ 
If[TypeX === Sinc && TypeY === Lagrange,
  TotSol[x_, y_, t_] :=  $\sum_{m=1}^{\text{NodesX}} \sum_{n=1}^{\text{NodesY}} (\text{aa}[[m, n, 2]][t] * \text{SincX}[m, x] * \text{LagrY}[n, y]);$ 
If[TypeX === Sinc && TypeY === Sinc, TotSol[x_, y_, t_] :=
   $\sum_{m=1}^{\text{NodesX}} \sum_{n=1}^{\text{NodesY}} (\text{aa}[[m, n, 2]][t] * \text{SincX}[m, x] * \text{SincY}[n, y]);$ 
Print["Solution:"];

Plot3D[Evaluate[TotSol[x, y, 1]], {x, 0, 1}, {y, 0, 1}, PlotPoints → 51,
  Boxed → True, AxesLabel → TraditionalForm /@ {x, y, "u(x,y)"},
  Ticks → {{0, 0.5, 1}, {0, 0.5, 1}, {0, 1}}, PlotRange → All]
];

{NodesX, NodesY} = {31, 31};
{CoeffA, CoeffB, LL} = {.1, 1, 40};
conciniz = Exp[-CoeffA * (LL (x - 0.5))2];
concBoundX0 = 0;
concBoundX1 = 0;
concBoundY0 = Exp[-CoeffA * (LL (x - 0.5))2] Exp[-CoeffB * t];
concBoundY1 = Exp[-CoeffA * (LL (x - 0.5))2] Exp[-CoeffB * t];
IBDataX = {concBoundX0, conciniz, concBoundX1};
IBDataY = {concBoundY0, conciniz, concBoundY1};
{TypeX, TypeY} = {Sinc, Lagrange};

SlabTwoDim[IBDataX, IBDataY, NodesX, NodesY, TypeX, TypeY]

```