Programming Finite Element Methods with B-Splines

K. Höllig and J. Hörner

Links:

FEMB program package $\longrightarrow$

http://www.siam.org/books/fr26/

B–spline demos $\longrightarrow$

http://www.siam.org/books/ot132/

Model Problems

partial differential equation of second order, Ritz-Galerkin approximation

$$Lu = f \quad \text{on} \quad D\,, \quad u = 0 \quad \text{on} \quad \Gamma \subseteq \partial D \quad , \quad a(u,v) = \int_D fv \quad \forall v \in \mathcal{H}$$

(P) Generalized Poisson problem

$$Lu = -\operatorname{div}(p \operatorname{grad} u) + qu$$

(E2) Two-dimensional elasticity (plane strain and plane stress)

$$Lu = -\alpha \begin{pmatrix} u^1_{xx} \\ u^2_{yy} \end{pmatrix} - (\beta + \gamma/4) \begin{pmatrix} u^2_{xy} \\ u^1_{xy} \end{pmatrix} - (\gamma/4) \begin{pmatrix} u^1_{yy} \\ u^2_{xx} \end{pmatrix}$$

(E3) Three-dimensional elasticity

$$Lu = -(q_1 + q_2) \begin{pmatrix} u^1_{xx} + u^2_{xy} + u^3_{xz} \\ u^1_{xy} + u^2_{yy} + u^3_{yz} \\ u^1_{xz} + u^2_{yz} + u^3_{zz} \end{pmatrix} - q_2 \begin{pmatrix} u^1_{xx} + u^1_{yy} + u^1_{zz} \\ u^2_{xx} + u^2_{yy} + u^2_{zz} \\ u^3_{xx} + u^3_{yy} + u^3_{zz} \end{pmatrix}$$

B-Splines

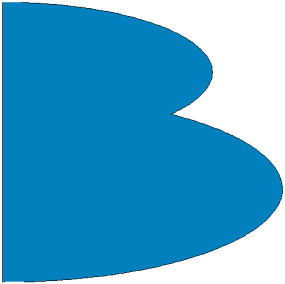
$$b_k(x) = b^n(x_1/h - k_1) \cdots b^n(x_d/h - k_d)$$

• nodes ξ^k, grid position k

- scaled translates of a single basis function
- one parameter per grid point
- arbitrary order and smoothness
- efficient and elegant algorithms

Weight Functions

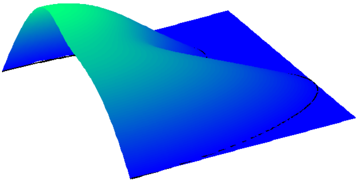
$D : w(x) > 0$



domain

Weight Functions

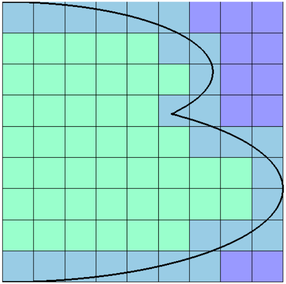
$D : w(x) > 0$



weight function

Weight Functions

$D : w(x) > 0$



grid

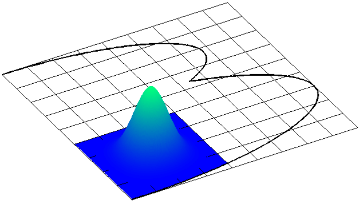
finite elements: $B_k = wb_k$

$$u_h = \sum_{k \in K} u_k B_k$$

K : rectangular array, irrelevant coefficients set to zero

Weight Functions

$D : w(x) > 0$



weighted B-splines

finite elements: $B_k = wb_k$

$$u_h = \sum_{k \in K} u_k B_k$$

K : rectangular array, irrelevant coefficients set to zero

Implementation

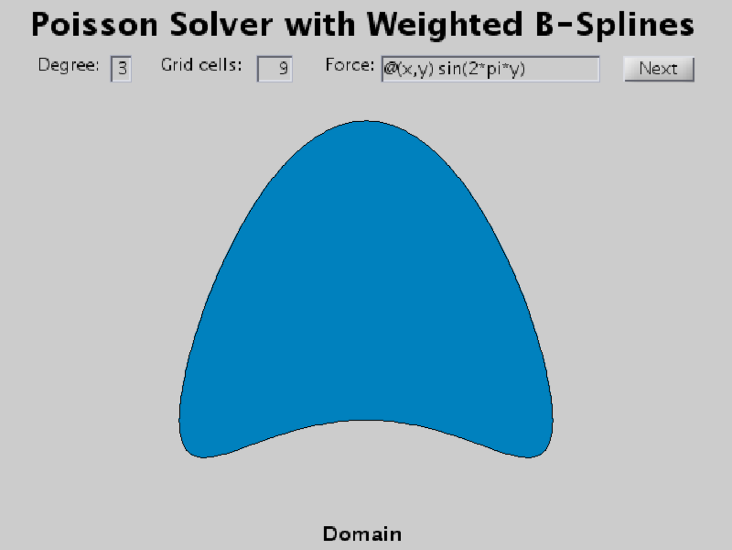
- select/construct weight function
- choose degree and grid width
- determine integration points
- assemble Ritz-Galerkin system
- iterative solution
- visualize weighted approximation
- compute residual

(the last five items: } FEMB)

sample program

```
% grid width, degree, algorithmic parameters (defaults)
H = 10; n = 3; PAR = set_par(H,n); PAR.values=15;
% domain (see m-file w_BASIC.m)
w = @w_BASIC;
% force
f = @(x,y) sin(y-x);
% finite element approximation and plot of solution
bvp_2d(@p_poisson,@q_poisson,f,w,w,H,n,PAR);
```

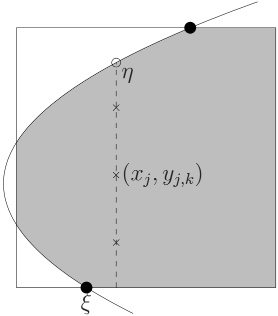
Implementation



Numerical Integration

$$\int_{Q\cap D} f \;=\; \int_0^h \underbrace{\left[\int_0^h f(x,y)\,\chi(x,y)\,dy\right]}_{=g(x)\text{ piecewise smooth}} dx$$

$$\approx \sum_j w_j g(x_j) \approx \sum_j w_j \left[\sum_k w_{j,k} f(x_j, y_{j,k})\right]$$

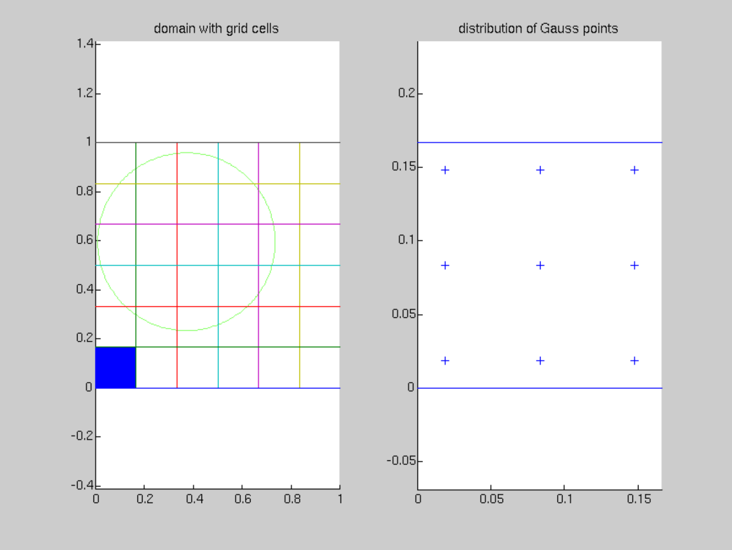


determine breakpoints

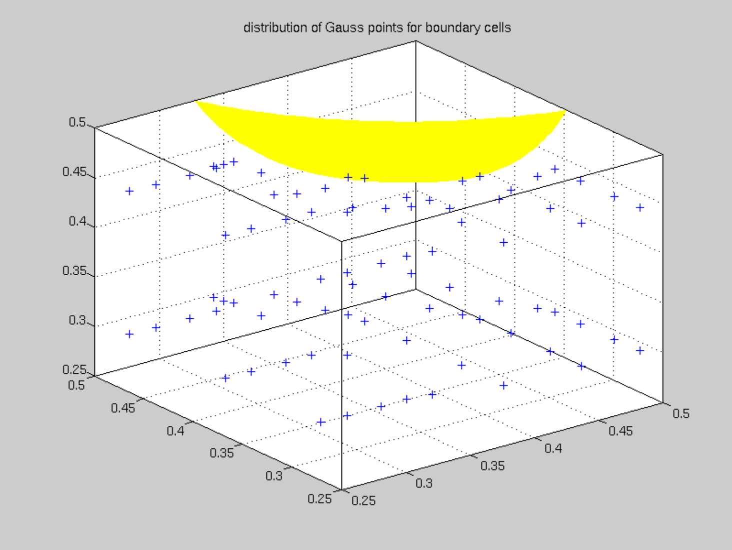
$$w \approx \sum\sum a_{j,k} x^j\, y^k$$

$$\xi : \quad y \in \{0, h\}, \quad \eta : x = x_j \longrightarrow \text{ roots of polynomials}$$

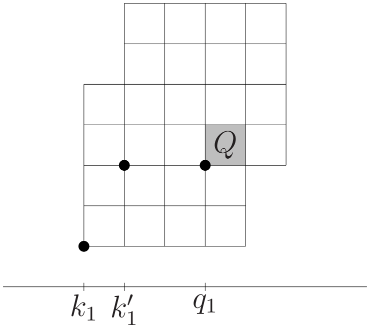
Numerical Integration (2D)



Numerical Integration (3D)



Matrix Assembly



for $q_1 = 1 : Q_1$, for $q_2 = 1 : Q_2$, ...
for $k_1 = q_1 - n : q_1$, for $k_2 = q_2 - n : q_2$, ...
for $k_1' = q_1 - n : q_1$, for $k_2' = q_2 - n : q_2$, ...
$G(k_1, k_2, \ldots, s_1, s_2, \ldots) \leftarrow$ add $\sum_\alpha w_\alpha^Q \varphi_{k,k'}^Q (x_\alpha^Q)$

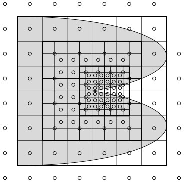
$s_\nu = k_\nu' - k_\nu$: B-spline offset
$\varphi_{k,k'}^Q$: integrand of the bilinear form on Q

Examples and Demos

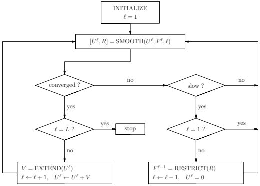
- `demo_spline_2d.m`
- `demo_spline_3d.m`
- `demo_integrate_2d.m`
- `demo_integrate_3d.m`
- `demo_poisson.m`
- `demo_bvp_2d.m`
- `demo_bvp_3d.m`
- `demo_elasticity_2d.m`
- `demo_elasticity_3d.m`

- example_BASIC.m
- example_bvp_2d_convergence.m
- example_bvp_3d_convergence.m
- example_elasticity_2d_disc.m
- example_elasticity_2d_tunnel.m
- example_elasticity_3d_dome.m
- example_bvp_3d_csg.m

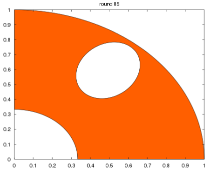
Next Steps



hierarchical bases



multigrid algorithms



(figure from F. Martin)

shape optimization