

Informatyka

Zmień przedmiot ▼

E-podręczniki

Moduły

Informacja

Strona zapisana (wersja 2).

Explicit method Brak plików do pobrania.

In the Euler method used to solve non-stationary problems, the differential operator describing the "physics" of the simulated phenomenon is computed at the previous time instant

$$\frac{u_{t+1}-u_t}{dt} - \mathcal{L}(u_t) = f_{t+1}$$

Thus, all parts of the equation are calculated on the basis of the state at the previous moment, and only the state correction at the current moment is performed:

$$u_{t+1} = u_t + dt * \mathcal{L}(u_t) + dt * f_{t+1}$$

We choose the so-called test functions w and we multiply our equation

$$(u_{t+1}, w) = (u_t + dt * \mathcal{L}(u_t) + dt * f_{t+1}, w)$$

Symbol (u, w) here stands for the dot product in space L^2 i.e. integration (while in the following formula the bracket means "ordinary" bracket)

$$\int_{\Omega} (u_{t+1} w) dx dy = \int_{\Omega} (u_t + dt * \mathcal{L}(u_t) + dt * f_{t+1}) w dx dy$$

Note that our problem is similar in structure to the bitmap projection problem

$$(u_{t+1}(x, y), w(x, y)) = (BITMAPA(x, y), w(x, y))$$

and in fact we can use the same code which, instead of reading the pixels of the bitmap, will sample the right side pixel by pixel.

So what exactly is our equation and the right side?

We will use a linear combination of the B-spline function to approximate the state of our system at a given moment in time. For this purpose, we select the basis of two-dimensional B-spline functions, specifying the node vectors in the direction of the axis of the coordinate system, for example, the two-dimensional basis of the second-order B-spline function

$$\{B_{i,2}^x(x)B_{j,2}^y(y)\}_{i=1,\dots,N_x;j=1,\dots,N_y} u_{i,j}$$

They will be used to approximate the simulated scalar field of the current time instant

$$u(x, y; t + 1) \approx \sum_{i=1,\dots,N_x;j=1,\dots,N_y} u_{i,j}^{t+1} B_i^x(x) B_j^y(y)$$

and at the previous moment in time

$$u(x, y; t) \approx \sum_{i=1,\dots,N_x;j=1,\dots,N_y} u_{i,j}^t B_i^x(x) B_j^y(y)$$

Similarly, we'll use the B-spline base functions for testing:

$$w(x, y) \in \{B_k^x(x)B_l^y(y)\}_{k=1,\dots,N_x;l=1,\dots,N_y} w^{k,l}$$

Assuming the differential operator \mathcal{L} describing "physics" is linear, our equation now looks like this:

$$\begin{aligned} & \sum_{i=1,\dots,N_x;j=1,\dots,N_y} u_{i,j}^{t+1} (B_i^x(x)B_j^y(y), B_k^x(x)B_l^y(y)) = \\ & = \sum_{i=1,\dots,N_x;j=1,\dots,N_y} u_{i,j}^t (B_i^x(x)B_j^y(y), B_k^x(x)B_l^y(y)) + \\ & + dt * \sum_{i=1,\dots,N_x;j=1,\dots,N_y} u_{i,j}^t \mathcal{L}(B_i^x(x)B_j^y(y), B_k^x(x)B_l^y(y)) + dt * (f_{t+1}(x, y), B_k^x(x)B_l^y(y)). \end{aligned}$$

$$\forall k = 1, \dots, N_x; l = 1, \dots, N_y$$

We do not establish a specific problem here, the conclusions presented here refer to any physical problem that can be simulated with the method described. For example, for the problem of heat transport we have

$$\begin{aligned} & \sum_{i=1,\dots,N_x;j=1,\dots,N_y} u_{i,j}^{t+1} (B_i^x(x)B_j^y(y), B_k^x(x)B_l^y(y)) = \\ & = \sum_{i=1,\dots,N_x;j=1,\dots,N_y} u_{i,j}^t (B_i^x(x)B_j^y(y), B_k^x(x)B_l^y(y)) + \\ & + dt * \sum_{i=1,\dots,N_x;j=1,\dots,N_y} u_{i,j}^t \left(\left(\frac{\partial^2 (B_i^x(x)B_j^y(y))}{\partial x^2} + dt * \frac{\partial^2 (B_i^x(x)B_j^y(y))}{\partial y^2} \right), B_k^x(x)B_l^y(y) \right) + \\ & + (f_{t+1}(x, y), B_k^x(x)B_l^y(y)) \quad \forall k = 1, \dots, N_x; l = 1, \dots, N_y \end{aligned}$$

that is, remembering that the symbol (u, w) here stands for the dot product in space L^2 i.e. integration, we have (in the next formula the bracket means "ordinary" bracket)

$$\sum_{i=1,\dots,N_x;j=1,\dots,N_y} u_{i,j}^{t+1} \int_{\Omega} B_i^x(x)B_j^y(y)B_k^x(x)B_l^y(y) dx dy =$$

$$\begin{aligned}
&= \sum_{i=1, \dots, N_x; j=1, \dots, N_y} u_{i,j}^t \int_{\Omega} B_i^x(x) B_j^y(y) B_k^x(x) B_l^y(y) dx dy + \\
&+ dt * \sum_{i=1, \dots, N_x; j=1, \dots, N_y} u_{i,j}^t \int_{\Omega} \left(\frac{\partial^2 (B_i^x(x) B_j^y(y))}{\partial x^2} + \frac{\partial^2 (B_i^x(x) B_j^y(y))}{\partial y^2} \right) B_k^x(x) B_l^y(y) dx dy + \\
&+ dt * \int_{\Omega} (f_{t+1}(x, y) B_k^x(x) B_l^y(y)) dx dy \quad \forall k = 1, \dots, N_x; l = 1, \dots, N_y
\end{aligned}$$

Coming back to the bitmap analogy, we have

$$\int_{\Omega} u_{t+1}(x, y) w(x, y) dx dy = \int_{\Omega} BITMAPA(x, y) w(x, y) dx dy$$

that is, our "bitmap function" is given

$$\begin{aligned}
BITMAPA(x, y) &= \sum_{i=1, \dots, N_x; j=1, \dots, N_y} u_{i,j}^t B_i^x(x) B_j^y(y) + \\
&+ \sum_{i=1, \dots, N_x; j=1, \dots, N_y} u_{i,j}^t dt \left(\frac{\partial^2 (B_i^x(x) B_j^y(y))}{\partial x^2} + \frac{\partial^2 (B_i^x(x) B_j^y(y))}{\partial y^2} \right) + dt * f_{t+1}(x, y)
\end{aligned}$$

and we can actually use the same code which instead of reading the pixels of the bitmap will sample the above right side "pixel by pixel".

Usually, on the right side, we can split the red term (assuming that the domain is large enough that the edge integral can be ignored because the modeled area or its derivatives are zero there)

$$\begin{aligned}
&\sum_{i=1, \dots, N_x; j=1, \dots, N_y} u_{i,j}^{t+1} \int_{\Omega} B_i^x(x) B_j^y(y) B_k^x(x) B_l^y(y) dx dy = \\
&= \sum_{i=1, \dots, N_x; j=1, \dots, N_y} u_{i,j}^t \int_{\Omega} B_i^x(x) B_j^y(y) B_k^x(x) B_l^y(y) dx dy + \\
&- dt * \sum_{i=1, \dots, N_x; j=1, \dots, N_y} u_{i,j}^t \int_{\Omega} \left(\frac{\partial (B_i^x(x) B_j^y(y))}{\partial x} + \frac{\partial (B_i^x(x) B_j^y(y))}{\partial y} \right) \left(\frac{\partial (B_k^x(x) B_l^y(y))}{\partial x} + \frac{\partial (B_k^x(x) B_l^y(y))}{\partial y} \right) dx dy + \\
&+ dt * \int_{\Omega} (f_{t+1}(x, y) B_k^x(x) B_l^y(y)) dx dy \quad \forall k = 1, \dots, N_x; l = 1, \dots, N_y
\end{aligned}$$

Note that the terms (red and blue) in which we calculate the values of the solution from the previous time, mean sampling the sum of all B-splines multiplied by the coefficients calculated in the previous time step. Such integration is performed according to the chapter "Variational formulation and numerical integration". In particular, note that these integrals can be broken down into the sum of the integrals of individual mesh elements, and on a given mesh element only $(p + 1)^2$ B-spline base functions, where p denotes the degree of the B-spline function.

Explicit method simulations are limited by the size of the time step. Increasing the spatial accuracy (increasing the mesh size of the finite element method to solve the right-to-left projection problem) requires reducing the size of the time step, otherwise

the simulation will start to behave unstably (colloquially the simulation will "explode"). This is expressed by the Courant-Friedrichs-Levi (CFL) condition.

$$\frac{u \Delta t}{\Delta x} < C_{max}$$

where u is the maximum value of the simulated phenomenon (traditionally it is the maximum speed, thanks to which the condition has a dimensionless form, which only applies to tasks with dominant convection and a constant or variable speed field), Δt is the length of the time step, Δx stands for element size (element diameter). The CFL condition was originally proposed for the finite difference method, however it also works for the finite element method.

Constant C_{max} cannot be exceeded by the quotient

$$\frac{u \Delta t}{\Delta x}.$$

If this happens (for example, the time step will be too large) then the simulation will start to behave unstably (colloquially speaking "it will explode") [1]

Below I present examples of two simulations of wave propagation for an elastic material, calculated using the explicit method. In the first case, the CFL condition is not met, in the second case it is met.

They have been numbered with the IGA-ADS code [2]

Niestabilnie

from [Centrum e-Learningu AGH](#)

00:16

Bibliografia

1. Richard Courant, Kurt Otto Friedrichs, Hans Lewy: Über die partiellen Differenzengleichungen der mathematischen Physik, *Mathematische Annalen* (in German), Germany 1928, dostęp:18.10.2019
2. Marcin Łoś, Maciej Woźniak, Maciej Paszyński, Andrew Lenharth, Muhammm Amber Hassan, Keshav Pingali: IGA-ADS: Isogeometric analysis FEM using ADS solver, *Computer & Physics Communications*, Elsevier 2017, dostęp:18.10.2019

Utworzona przez [admin](#). Ostatnia aktualizacja: Wtorek 03 z Listopad, 2020 23:03:42 UTC przez paszynsk@agh.edu.pl. Autor: Maciej Paszynski

STATUS: **W opracowaniu**

[Zgłoś do recenzji](#)

[Edytuj](#)

[Jak to działa?](#)

[O e-podręcznikach AGH](#)

[Regulamin](#)

[Polityka prywatności](#)

[Licencja CC BY-SA](#)

[Partnerzy](#)

[Kontakt](#)

[Prześlij opinię](#)

[About](#)



OpenAGH e-podręczniki | Explicit method
Akademia Górniczo-Hutnicza
im. Stanisława Staszica w Krakowie
Centrum e-Learningu

Centrum e-Learningu AGH ©2013–2020

Wersja mobilna