ATM: Automated Testing for Numerical Simulation Program via Derivative Invariance

ANONYMOUS AUTHOR(S)*

ACM Reference Format:
Anonymous Author(s). 2024. ATM: Automated Testing for Numerical Simulation Program via Derivative Invariance. 1, 1 (September 2024), 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

A Soundness proof of PDEs

(1) One initial condition case We take the heat equation as an instance and we consider the second time step. Since u_1^2 and $u_{N_x-1}^2$ are the function f of our method, we have

$$u_1^2 = k(u_2^1 - 2u_1^1 + u_0^1) + u_1^1$$
$$u_{N_x-1}^2 = k(u_{N_x}^1 - 2u_{N_x-1}^1 + u_{N_x-2}^1) + u_{N_x-1}^1,$$

where we omit the term f_i^n and express $\alpha\Delta t/\Delta x^2$ with k for the sake of conciseness. As for $u_2^1, u_1^1, u_{N_x-1}^1, u_{N_x-2}^1$, we have

$$u_2^1 = k(u_3^0 - 2u_2^0 + u_1^0) + u_2^0$$
$$u_1^1 = k(u_2^0 - 2u_1^0 + u_0^0) + u_1^0$$
$$u_{N_x-1}^1 = k(u_{N_x}^0 - 2u_{N_x-1}^0 + u_{N_x-2}^0) + u_{N_x-1}^0$$
$$u_{N_x-2}^1 = k(u_{N_x-1}^0 - 2u_{N_x-2}^0 + u_{N_x-3}^0) + u_{N_x-2}^0$$

and $u_0^1, u_{N_x}^1$ depend on the boundary conditions. Regarding the Neumann conditions, they set values of $\partial u/\partial n$, such as $\partial u/\partial n = 0$, where $u_0^1 = u_2^1$ and $u_{N_x}^1 = u_{N_x-2}^1$. As for the periodic condition, we would have $u_0^1 = u_{N_x-2}^1$ and $u_{N_x}^1 = u_2^1$. Therefore, to check the boundary conditions, we could alter the valus of $u_0^0, u_1^0, u_2^0, u_3^0, u_{N_x-3}^0, u_{N_x-2}^0, u_{N_x-1}^0, u_{N_x}^0$ to compute the numerical derivatives of u_1^2 and $u_{N_x-1}^2$. Since these independent variables (*i.e.*, the values of u of the initial condition) could cover all instants in the iteration, ATM could ensure the soundness of **the core code block**.

(2) Two initial conditions case We use the wave equation as an illustrative example, where we take $\{u_1^2, u_{N_x-1}^2, u_1^3, u_{N_x-1}^3\}$ as the functions f. Regarding the second time step, we have

$$u_1^2 = (2-2g)u_1^1 - u_1^0 + gu_2^1 + gu_0^1$$
$$u_{N_x-1}^2 = (2-2g)u_{N_x-1}^1 - u_{N_x-1}^0 + gu_{N_x}^1 + gu_{N_x-2}^1,$$

where

$$u_2^1 = u_2^0 + 0.5g(u_3^0 - 2u_2^0 + u_1^0)$$
$$u_1^1 = u_1^0 + 0.5g(u_2^0 - 2u_1^0 + u_0^0)$$
$$u_{N_x-1}^1 = u_{N_x-1}^0 + 0.5g(u_{N_x}^0 - 2u_{N_x-1}^0 + u_{N_x-2}^0)$$
$$u_{N_x-2}^1 = u_{N_x-2}^0 + 0.5g(u_{N_x-1}^0 - 2u_{N_x-2}^0 + u_{N_x-3}^0)$$

and $u_0^1, u_{N_x}^1$ depend on the boundary condition. We abbreviate $c^2\Delta t^2/\Delta x^2$ as g. In a manner akin to the **one initial condition case**, ATM could test all parts of **the core code block** and the boundary condition of the computation for the first step if we alter the value of $\{u_0^0, u_1^0, u_2^0, u_3^0, u_{N_x-3}^0, u_{N_x-2}^0, u_{N_x-1}^0, u_{N_x}^0\}$ and calculate the numerical derivatives, respectively. If all parts above are correct, we

2024. ACM XXXX-XXXX/2024/9-ART
https://doi.org/10.1145/nnnnnnn.nnnnnnn

utilize the third time step $\{u_1^3, u_{N_x-1}^3\}$ to test the boundary condition of **the main iteration**, which is the last portion. We now have

$$u_1^3 = (2-2g)u_1^2 - u_1^1 + gu_2^2 + gu_0^2$$
$$u_{N_x-1}^3 = (2-2g)u_{N_x-1}^2 - u_{N_x-1}^1 + gu_{N_x}^2 + gu_{N_x-2}^2$$

where

$$\begin{aligned}
u_2^2 &= (2g-2g^2)u_3^0 + (3g^2-4g+1)u_2^0 + \\
&\quad (2g-2g^2)u_1^0 + 0.5g^2u_4^0 + 0.5g^2u_0^0 \\
u_1^2 &= (g-g^2)u_0^0 + (2.5g^2-4g+1)u_1^0 + \\
&\quad (-2g^2+2g)u_2^0 + 0.5g^2u_3^0 + gu_0^1 \\
u_{N_x-2}^2 &= (2g-2g^2)u_{N_x-1}^0 + (3g^2-4g+1)u_{N_x-2}^0 + \\
&\quad (2g-2g^2)u_{N_x-3}^0 + 0.5g^2u_{N_x}^0 + 0.5g^2u_{N_x-4}^0 \\
u_{N_x-1}^2 &= (g-g^2)u_{N_x}^0 + (2.5g^2-4g+1)u_{N_x-1}^0 + \\
&\quad (-2g^2+2g)u_{N_x-2}^0 + 0.5g^2u_{N_x-3}^0 + gu_{N_x}^1
\end{aligned}$$

and $\{u_0^1, u_{N_x}^1, u_0^2, u_{N_x}^2\}$ depend on the boundary condition. In a similar manner, $\{u_0^0, u_1^0, u_2^0, u_3^0, u_4^0, u_{N_x-4}^0, u_{N_x-3}^0, u_{N_x-2}^0, u_{N_x-1}^0, u_{N_x}^0\}$ are independent variables of u_1^3 and $u_{N_x-1}^3$. If we alter these variables and calculate the numerical derivatives, we could detect the bug of boundary condition which specify in **the main iteration**.