Solving Inverse Heat Transfer Problems: Recovery of Source Strength and Temperature Field from Sparse Observations

Steffen Wedig

January 3, 2024

1 Introduction

In this sample of work, I investigate a inverse, one-dimensional heat transfer problem. Inverse problems refer to mathematical problems that aim to discover model parameters based on model outputs. On the contrary, direct problems calculate model results based on parameters. In the following, the goal is to recover the full temperature field and the unknown heat source strength from sparse temperature measurements. The heat equation describes this problem:

$$\frac{\partial T}{\partial t} = \alpha \nabla^2 T + f(x,t) \quad \text{for } a \leq x \leq b,\ 0 \leq t \leq T, \tag{1}$$

where $T(x,t)$ represents the temperature field at position x and time t, α is the constant diffusivity, $f(x,t)$ is the spatially and temporally varying source strength. Zero Neumann boundary conditions are defined for both ends of the domain, as well as the initial conditions.

$$\frac{\partial T}{\partial x}(x,t) = 0 \qquad \text{at } x = a, b, \tag{2}$$

$$T(x,0) = T_0(x) \qquad \text{for } a < x < b. \tag{3}$$

In the simulation, the temperature field $T(x,t)$ is observed sparsely; the temperature is measured only at a given number of sensors, but without noise.

In this sample of work, this inverse problem is solved using differentiable numerical simulations. Differentiable numerical simulations combine traditional numerical methods and the automatic differentiation capabilities of deep learning frameworks. In this case, the finite difference method provides a discrete formulation $P(T,\nu)$ that maps the current state $T(x,t)$ to a future state of the system $T(x,t+\Delta t)$ using the model parameters ν. The discrete formulation consists of a sequence of operations, and the derivatives with respect to the input can be obtained by using the chain rule on this sequence of operations. The derivatives enable the iterative updating of the model parameters so that the difference between the simulation results and the sensor measurements is minimised [2].

2 Methodology

Equation 1 is solved numerically with the finite difference method, specifically the 2nd order Crank Nicholson scheme. The discretised formulation of the heat equation yields a system of equation, which is solved for the temperature at time index $n+1$ for all spatial points index i, namely

$$\frac{T_i^{n+1} - T_i^n}{\Delta t} = \frac{\alpha}{2}\left(\frac{T_{i+1}^{n+1} - 2T_i^{n+1} + T_{i-1}^{n+1}}{\Delta x^2} + \frac{T_{i+1}^n - 2T_i^n + T_{i-1}^n}{\Delta x^2}\right). \tag{4}$$

The spatial domain is discretised between $a=-1, b=1$ in $N_x = 200$ gridpoints; time is discretised between $t=0$ and $t=1$ in $N_t = 200$ gridpoints as well. Implementing this in the deep learning framework *jax* allows the use of automatic differentiation [1]. A bonus of using *jax* is the possibility to speed up the simulation by using just-in-time compilation, and vectorisation of operations. *Jax* also supports using hardware accelerators such as GPUs out of the box.

The following steps solve the inverse problem:

1. Compute a reference trajectory of sensor measurements with a reference heat source.
2. Formulate an initial guess for the heat source.
3. Compute a test trajectory of temperature measurements using the finite difference method.
4. Calculate a loss between the reference and test trajectory.
5. Differentiate the loss function wrt. the test source strength.
6. Update the test source strength by taking an optimisation step.
7. Repeat steps 3-6 for a given number of steps.

After the full reference trajectory is calculated, only a certain percentage of the domain is "measured" by slicing the complete reference trajectories at given, equidistant sensor locations, which yields "sensor trajectories" (Step 1). The sensors are placed at ten equidistant locations, covering 5% of the grid. All sensor measurements yield temperatures in arbitrary units. The initial guess for the source field is valued at 1 for all gird points (Step 2). For all trajectories, the initial temperature distribution is known in the whole domain. A test trajectory is calculated from the guess and the initial temperature and measured at the sensor positions(Step 3). The loss of each trajectory is calculated as the L^2 norm between the test and reference sensor trajectory (Step 4). The gradient wrt. to the source strength and diffusivity is calculated automatically and fed into the Adam optimiser (Step 5). All experiments use a learning rate of 0.005 for the parameter updates (Step 6). Steps 3-6 are repeated for 500 epochs.

2.1 Description of Test Cases

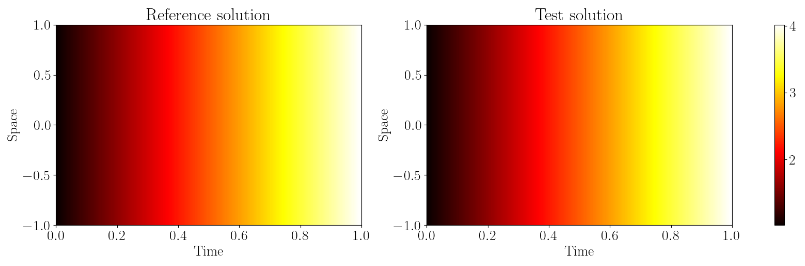
Each of the three different test cases investigates a different source field. The source functions in each experiment are given as follows:

$$\begin{aligned} f_1(x,t) &= 3 \\ f_2(x,t) &= \begin{cases} 3, & \text{if } -0.3 \le x \le 0.3 \\ 0, & \text{otherwise} \end{cases} \\ f_3(x,t) &= e^{-\frac{(x-0.25\sin(t))^2}{0.3}} \end{aligned} \tag{5}$$

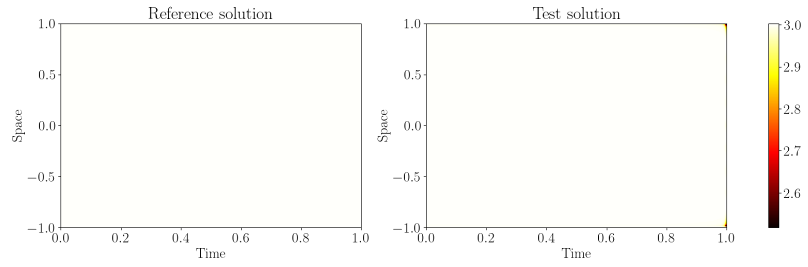
The first source function is constant in space and time. The second source function is constant in time and piece-wise constant in space. A discontinuity exists at $x=-0.3$ and $x=0.3$. The third source term is a Gaussian function, with the centre moving sinusoidally.

3 Results and Discussion

Figures 1 to 3 show the results of the differentiable physics simulation. Each figure compares the reference to the test solution, first for the temperature field and then the source field. Qualitatively, the temperature field is recovered well in all cases.



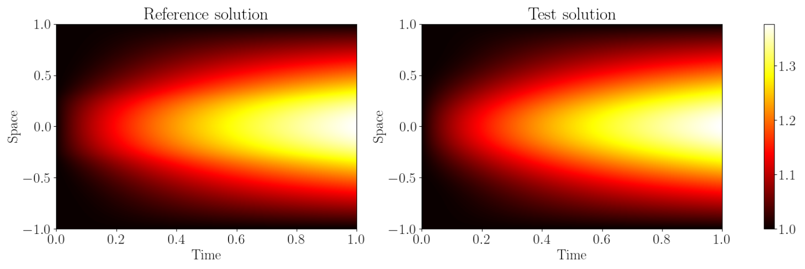
(a) Reference Temperature Field and the Final Temperature Field Solution.



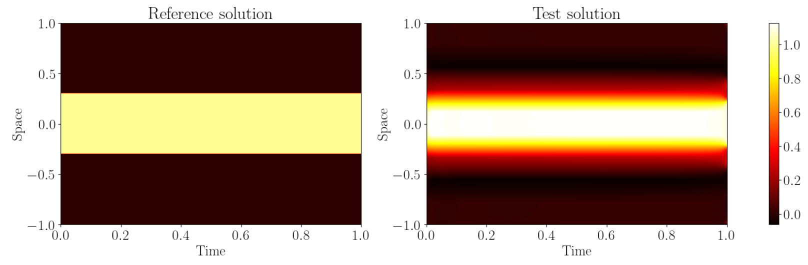
(b) Reference Source Field and the Final Source Field Solution.

Figure 1: Comparison of Reference and Test Solution for the Constant Test Case.

The source field is recovered almost perfectly in the constant test case, as shown in Figure 1. Errors in the source field exist only at the very end of the trajectory. The source values there do not impact the temperature measurements, as the heat does not diffuse to the sensor locations before the end of the simulation, thus not increasing the loss value.



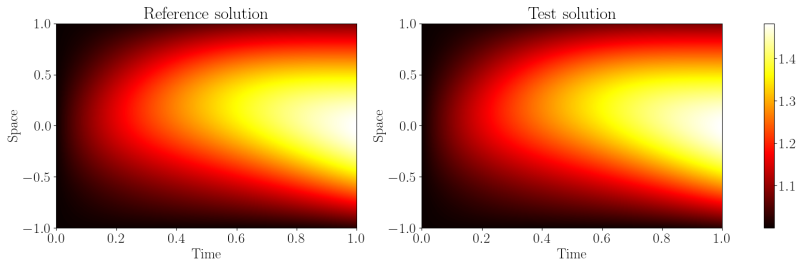
(a) Reference Temperature Field and the Final Temperature Field Solution.



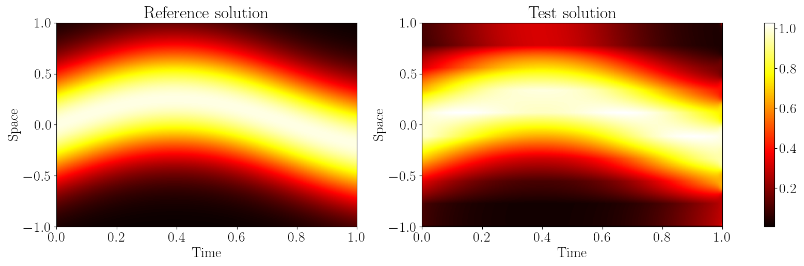
(b) Reference Source Field and the Final Source Field Solution.

Figure 2: Comparison of Reference and Test Solution for the Piece-wise Test Case.

For the piece-wise test case, the heat source recovery is more inaccurate due to the discontinuities in the source function. Information on the source location is lost as the heat diffuses away from the source. This test case illustrates the ill-posedness of inverse problems well. Different source fields can produce very similar temperature measurements. Based on the sparse temperature measurements alone, it is difficult to find the reference solution because the difference between the proposed solution and the reference temperature field is small.



(a) Reference Temperature Field and the Final Temperature Field Solution.



(b) Reference Source Field and the Final Source Field Solution.

Figure 3: Comparison of Reference and Test Solution for the Gaussian Test Case.

The moving Gaussian source function is recovered well in the third test case. Reference and test solutions are similar for the temperature and the source field. At the sensor locations, some artefacts exist in the test solution of the source field. Rerunning the simulation with a higher number of sensors reduces the artefacts.

Table 1 quantitatively shows the loss minimisation for the temperature and source field. During the optimisation, the loss is only calculated at the sensor locations. Below, the loss is given for all grid points over the whole domain for evaluation. The table compares the loss for the initial guess and the loss at the final iteration step. In all test cases, the temperature loss is reduced drastically and reaches a final value close to zero, showing that it is possible to recover the entire temperature field from sparse measurement. While the solver can reduce the source loss, a significant error remains after 500 iterations in the second and third test cases.

Test Case	$L_{T,ini}$	$L_{T,end}$	$L_{S,ini}$	$L_{S,end}$
Constant	232.97	0.055	400	0.874
Piece-wise	91.26	0.24	176.33	32.86
Gaussian	78.56	0.04	125.079	5.166

Table 1: Comparison of the Initial and Final Temperature Loss ($L_{T,ini}$, $L_{T,end}$) and Initial and Final Source Loss for the whole domain ($L_{S,ini}$, $L_{S,end}$) in arbitrary units.

Multiple approaches exist to reduce the error further. First, the number of sensors can be increased, which provides a more accurate observation of the temperature field. In practice, it is desirable to use methods that are robust to sparse measurements and require only as few sensors as possible. Second, the number of degrees of freedom can be reduced. In this sample of work, the heat source values could be altered independently at each gridpoint, meaning that there are $N_t \cdot N_x$ degrees of freedom. However, the reference source field is calculated from a given function. Therefore, the optimisation could be simplified by approximating the function instead of the values of the function at all grid points. The heat source function could either be approximated in a general fashion by using neural networks or by prescribing a specific functional form with fewer parameters. The second approach also allows the method to include more prior knowledge. To give a particular example, In laser-based additive manufacturing, the beam shape is known to be Gaussian, and the position of the beam centre is known because it is programmed in advance. The absorptivity of a new material might be the only unknown. By only updating the absorptivity, the degrees of freedom are drastically reduced.

4 Conclusion and Outlook

In this work sample, differentiable numerical simulations solved three inverse heat transfer problems. While the full temperature field was readily recovered from sparse measurements, the source field's reconstruction proved difficult in cases where the source function was transient and contained discontinuities.

In the future, I plan to further expand the scope of the project:

1. I want to include 2D and 3D problems, because they are more applicable in practice. One particular problem I want to tackle is measuring the temperature only on a domains boundary and recovering the temperature in the inner part of the domain, akin to observing the surface temperature of a powder bed in additive manufacturing processes.
2. I plan to adapt the solver to recover variable conductivity in addition to the heat source and temperature field. Discretising the heat equation with variable conductivity yields a different system of equations to be solved. Therefore, the finite difference scheme needs to be updated to approach this problem.
3. Instead of independently varying the heat source values at all grid points, I plan to use neural networks to approximate the heat source function.

5 Code Availability

The code for this sample of work is available on `https://github.com/Steffen195/InverseHeatEq`.

References

[1] James Bradbury et al. *JAX: composable transformations of Python+NumPy programs.* Version 0.3.13. 2018. URL: `http://github.com/google/jax`.

[2] Nils Thuerey et al. *Physics-based Deep Learning.* 2021. URL: `physicsbaseddeeplearning.org`.