PDEs and Boundary Conditions

New methods have been implemented for solving partial differential equations with boundary condition (PDE and BC) problems.

1st order PDE with a single boundary condition (BC) that does not depend on the independent variables

The PDE & BC project, started five years ago implementing some of the basic methods found in textbooks to match arbitrary functions and constants to given PDE boundary conditions of different kinds. One frequent problem is that of a 1st order PDE that can be solved *without boundary conditions* in terms of an arbitrary function, and where a single boundary condition (BC) is given for the PDE unknown function, and this BC does not depend on the independent variables of the problem. The problem can be solved making simple, however, ingenious use of differential invariants to match the boundary condition.

The examples that can now be handled using this new method, although restricted in generality to "***only one 1st order linear or nonlinear PDE and only one boundary condition for the unknown function itself***", illustrate well how powerful it can be to use more advanced methods.

First consider a linear example, among the simplest one could imagine:

```
> PDEtools:-declare(f(x, y, z))
```

$$f(x, y, z) \text{ will now be displayed as } f \tag{1.1}$$

```
> pde := diff(f(x, y, z), x) + diff(f(x, y, z), y) + diff(f(x, y, z), z) = f(x, y, z)
```

$$pde := f_x + f_y + f_z = f \tag{1.2}$$

Now, input a boundary condition (BC) for the unknown $f(x, y, z)$ such that this BC does not depend on the independent variables $\{x, y, z\}$; this BC can, however, depend on arbitrary symbolic parameters. For instance:

```
> bc := f(α + β, α - β, 1) = α·β
```

$$bc := f(\alpha + \beta, \alpha - \beta, 1) = \alpha\,\beta \tag{1.3}$$

This kind of problem can now be solved in one step:

```
> sol := pdsolve([pde, bc])
```

(1.4)

$$sol := f = \frac{(x - 2z + 2 + y)\,(x - y)\,e^{z-1}}{4} \tag{1.4}$$

To verify this result for correctness, use the pdetest. This also tests the solution against the boundary conditions.

> *pdetest*(*sol*, [*pde*, *bc*])

$$[0, 0] \tag{1.5}$$

To obtain the solution **(1.4)**, the PDE was first solved regardless of the boundary condition:

> *pdsolve*(*pde*)

$$f = \_F1(-x + y, -x + z)\,e^{x} \tag{1.6}$$

Next, the arbitrary function $\_F1(-x+y, -x+z)$ was determined such that the boundary condition $f(\alpha+\beta, \alpha-\beta, 1) = \alpha\beta$ is matched. Concretely, the *mapping* *_F1* is what was determined. You can see this mapping *reversing* the solving process in two steps. Start by taking the difference between the general solution **(1.6)** and solution **(1.4)** that matches the boundary condition:

> **(1.6)**-**(1.4)**

$$0 = \_F1(-x + y, -x + z)\,e^{x} - \frac{(x - 2z + 2 + y)\,(x - y)\,e^{z-1}}{4} \tag{1.7}$$

and isolate here $\_F1(-x+y, -x+z)$

> *PDEtools*:-*Solve*(**(1.7)**, $\_F1(-x+y, -x+z)$)

$$\_F1(-x + y, -x + z) = \frac{e^{-x+z-1}\left(x^2 - 2zx - y^2 + 2zy + 2x - 2y\right)}{4} \tag{1.8}$$

So this is the value that $\_F1(-x+y, -x+z)$ determined. To see the actual solving mapping _F1, that takes for arguments $-x+y$ and $-x+z$ and returns the right-hand side of **(1.8)**, one can perform a change of variables introducing the two parameters τ_1 and τ_2 of the *_F1* mapping:

> $\{\tau_1 = y - x, \tau_2 = -x + z, \tau_3 = z\}$

$$\{\tau_1 = -x + y, \tau_2 = -x + z, \tau_3 = z\} \tag{1.9}$$

> *solve*(**(1.9)**, $\{x, y, z\}$)

$$\{x = -\tau_2 + \tau_3, y = -\tau_2 + \tau_1 + \tau_3, z = \tau_3\} \tag{1.10}$$

> *PDEtools*:-*dchange*(**(1.10)**, **(1.8)**, $u \to simplify(u, size)$)

$$\_F1(\tau_1, \tau_2) = -\frac{e^{\tau_2 - 1}\,\tau_1\left(\tau_1 - 2\tau_2 + 2\right)}{4} \tag{1.11}$$

So, the solving mapping _F1 is:

> $\_F1 = unapply(rhs(\textbf{(1.11)}), \tau_1, \tau_2)$

$$\_F1 = (\tau_1, \tau_2) \mapsto -\frac{e^{\tau_2 - 1}\,\tau_1\left(\tau_1 - 2\tau_2 + 2\right)}{4} \tag{1.12}$$

Although this PDE and BC example looks simple, this solution **(1.12)** is not apparent, as is the way to get it just from the boundary condition

$f(\alpha+\beta, \alpha-\beta, 1) = \alpha\cdot\beta$ and the solution **(1.6)**.

Skipping the technical details, the key observation to compute a solving mapping is that: Given a 1st order PDE, where the unknown depends on k independent variables, if the boundary condition depends on $k-1$ arbitrary symbolic parameters α, β, one can always seek a "*relationship between these* $k-1$ *parameters and the* $k-1$ *differential invariants that enter as arguments in the arbitrary function _F1 of the solution*", and get the form of the mapping _F1 from this relationship and the BC. The method works in general. However, if, for instance, we change the BC **(1.3)**, making its right-hand side a sum instead of a product,

> $bc := f(\alpha+\beta, \alpha-\beta, 1) = \alpha+\beta$

$$bc := f(\alpha+\beta, \alpha-\beta, 1) = \alpha+\beta \quad \textbf{(1.13)}$$

> $sol := pdsolve([pde, bc])$

$$sol := f = (x-z+1)\, e^{z-1} \quad \textbf{(1.14)}$$

> $pdetest(sol, [pde, bc])$

$$[0, 0] \quad \textbf{(1.15)}$$

an interesting case happens when the boundary condition depends on less than $k-1$ parameters. For instance:

> $bc_1 := subs(\beta = \alpha, bc)$

$$bc_1 := f(2\,\alpha, 0, 1) = 2\,\alpha \quad \textbf{(1.16)}$$

> $sol_1 := pdsolve([pde, bc_1])$

$$sol_1 := f = \frac{((x-z+1)\,\_C1 + x - y)\, e^{\frac{(z-1)\,\_C1 + y}{1+\_C1}}}{1+\_C1} \quad \textbf{(1.17)}$$

As we see in this result, the additional difficulty represented by having few parameters got tackled by introducing an arbitrary constant _C1 (this is likely to evolve into something more general...)

> $pdetest(sol_1, [pde, bc_1])$

$$[0, 0] \quad \textbf{(1.18)}$$

Finally, consider a nonlinear example:

> $PDEtools:-declare(u(x, y))$

$$u(x, y) \text{ will now be displayed as } u \quad \textbf{(1.19)}$$

> $pde := 3\,(u(x, y) - y)^2 \left(\frac{\partial}{\partial x} u(x, y)\right) - \left(\frac{\partial}{\partial y} u(x, y)\right) = 0$

$$pde := 3\,(u-y)^2 u_x - u_y = 0 \quad \textbf{(1.20)}$$

Here we have two independent variables, so for illustration purposes use a boundary condition that depends on only one arbitrary parameter.

> $bc := u(0, \alpha) = \alpha$

$$bc := u(0, \alpha) = \alpha \quad \textbf{(1.21)}$$

All looks OK, but we still have another problem: check the arbitrary function _F1 entering the general solution of PDE when tackled without any boundary

condition:

> $pdsolve(pde)$

$$u = RootOf\left(-y^3 + 3\,y^2\,\_Z - 3\,y\,\_Z^2 + \_Z^3 - \_F1(\_Z) - x\right) \tag{1.22}$$

Remove this **RootOf** to see the underlying algebraic expression:

> $DEtools[remove\_RootOf]\,(\mathbf{(1.22)})$

$$-y^3 + 3\,y^2\,u - 3\,y\,u^2 + u^3 - \_F1(u) - x = 0 \tag{1.23}$$

So this is a PDE where the general solution is implicit, actually depending on an arbitrary function of the unknown $u(x, y)$. The code handles this problem in the same way, just that in cases like this there may be more than one solution. For this very particular BC **(1.21)**, there are actually three solutions:

> $pdsolve([pde, bc])$

$$u = x^{1/3} + y,\ u = -\frac{x^{1/3}}{2} - \frac{I\sqrt{3}\,x^{1/3}}{2} + y,\ u = -\frac{x^{1/3}}{2} + \frac{I\sqrt{3}\,x^{1/3}}{2} + y \tag{1.24}$$

Verify these three solutions against the PDE and the boundary condition.

> $map(pdetest, [\mathbf{(1.24)}], [pde, bc])$

$$[[0, 0], [0, 0], [0, 0]] \tag{1.25}$$

Linear PDE on bounded domains with homogeneous boundary conditions

More PDE on bounded domains are solved in Maple 2016.

Example: The wave equation

> $pde := diff(u(x, t), t, t) = c\text{^}2 * (diff(u(x, t), x, x));$

$$pde := \frac{\partial^2}{\partial t^2}\,u(x, t) = c^2\left(\frac{\partial^2}{\partial x^2}\,u(x, t)\right) \tag{2.1}$$

governs the displacements of a string whose length is l, so that $0 \le x \le l$, and $t \ge 0$.

> $bc := u(0, t) = 0, u(l, t) = 0;$

$$bc := u(0, t) = 0, u(l, t) = 0 \tag{2.2}$$

> $pdsolve([pde, bc])\ \text{assuming}\ l > 0, x \le l;$

$$u(x, t) = \_C4\,\sin\left(\frac{\pi\,x\,|\_ZI|}{l}\right)\left(\_C2\,\sin\left(\frac{c\,\pi\,t\,|\_ZI|}{l}\right) + \_C3\,\cos\left(\frac{c\,\pi\,t\,|\_ZI|}{l}\right)\right) \tag{2.3}$$

Many of the improvements were made when using the Fourier method (with separation of variables by product and eigenfunction expansion). This method separates the PDE by product into two ODEs, so that we now need to solve two ODE boundary problems. One of these ODE boundary problems is a Sturm-Liouville problem (an eigenvalue problem), whose solution we represent using an infinite series.

Example: Consider the diffusion PDE and BC problem below, where $0 \le x \le l$, $t \ge 0$.

> $pde := diff(u(x,t),t) = k*(diff(u(x,t),x,x));$

$$pde := \frac{\partial}{\partial t} u(x,t) = k\left(\frac{\partial^2}{\partial x^2} u(x,t)\right) \tag{2.4}$$

> $bc := (\mathrm{D}[1](u))(0,t) = 0, u(l,t) = 0, u(x,0) = f(x);$

$$bc := \mathrm{D}_1(u)(0,t) = 0, u(l,t) = 0, u(x,0) = f(x) \tag{2.5}$$

> $pdsolve([pde, bc])$ assuming $l > 0, x \le l;$

$$u(x,t) \tag{2.6}$$

$$= \sum_{\_Z3=1}^{\infty} \frac{1}{l}\left(2\,\mathrm{e}^{-\frac{\pi^2 (2\_Z3+1)^2 k t}{4 l^2}} \cos\left(\frac{\pi (2\_Z3+1) x}{2 l}\right)\left(\int_0^l f(x) \cos\left(\frac{\pi (2\_Z3+1) x}{2 l}\right) dx\right)\right)$$

Example: Consider Laplace's equation on a bounded circular domain. The PDE and BC problem below is in polar coordinates, with $0 \le r \le R$, $0 \le \theta \le 2\pi$.

> $pde := diff(diff(u(r,\mathrm{theta}),r),r) + 1/r*diff(u(r,\mathrm{theta}),r) + 1/r\text{^}2*diff(diff(u(r,\mathrm{theta}),\mathrm{theta}),\mathrm{theta}) = 0;$

$$pde := \frac{\partial^2}{\partial r^2} u(r,\theta) + \frac{\frac{\partial}{\partial r} u(r,\theta)}{r} + \frac{\frac{\partial^2}{\partial \theta^2} u(r,\theta)}{r^2} = 0 \tag{2.7}$$

> $bc := u(R,\mathrm{theta}) = f(\mathrm{theta}), u(r,0) = u(r, 2*\mathrm{Pi}), \mathrm{D}[2](u)(r,0) = \mathrm{D}[2](u)(r, 2*\mathrm{Pi});$

$$bc := u(R,\theta) = f(\theta), u(r,0) = u(r, 2\pi), \mathrm{D}_2(u)(r,0) = \mathrm{D}_2(u)(r, 2\pi) \tag{2.8}$$

> $pdsolve([pde, bc])$ assuming $r \le R, R > 0, \mathrm{theta} \ge 0, \mathrm{theta} \le 2\cdot\mathrm{Pi};$

$$u(r,\theta) = \frac{\_C10}{2} + \left(\sum_{\_Z5=1}^{\infty} \frac{1}{\pi}\left(r^{\_Z5}\left(\cos(\_Z5\,\theta)\left(\int_0^{2\pi} f(\theta)\cos(\_Z5\,\theta)\,d\theta\right) + \left(\int_0^{2\pi} f(\theta)\sin(\_Z5\,\theta)\,d\theta\right)\sin(\_Z5\,\theta)\right) R^{-\_Z5}\right)\right) \tag{2.9}$$

• Problems that include a source term can now be solved as well.

Example: Consider the following inhomogeneous PDE and BC problem, where the PDE includes a source term. To solve this, we use Duhamel's principle, namely that the solution to our inhomogeneous PDE problem can be found by solving the homogeneous version of the problem for a new variable $w(x, t, \tau)$,

such that

$u(x, t) = \int_0^t w(x, t-\tau, \tau)\, d\tau$, $w(x, 0, \tau) = f(x, \tau)$. The domain is bounded: $k > 0$, $0 \le x \le \mathrm{Pi}$, $t \ge 0$.

```
> pde := diff(u(x,t),t) - k*diff(u(x,t),x,x) = f(x,t);
```

$$pde := \frac{\partial}{\partial t} u(x, t) - k \left(\frac{\partial^2}{\partial x^2} u(x, t) \right) = f(x, t) \tag{2.10}$$

```
> bc := u(0,t) = 0, u(Pi,t) = 0, u(x,0) = 0;
```

$$bc := u(0, t) = 0, u(\pi, t) = 0, u(x, 0) = 0 \tag{2.11}$$

```
> pdsolve([pde, bc]) assuming k > 0, x >= 0, x <= Pi;
```

$$u(x, t) = \int_0^t \left(\sum_{\_Z6=1}^{\infty} \frac{2 \left(\int_0^{\pi} f(x, \tau 1) \sin(\_Z6\, x)\, dx \right) \sin(\_Z6\, x)\, e^{-\_Z6^2 k (t-\tau 1)}}{\pi} \right) d\tau 1 \tag{2.12}$$

Cauchy problem for hyperbolic PDE with or without sources

The method for solving the Cauchy problem for hyperbolic PDE (in unbounded domains) has been expanded to include different types of sources as well as functions in the initial conditions.

Here is an example without sources in the PDE, where now we can have different types of functions in the initial conditions.

```
> pde := diff(u(x,t),t,t) - 4*diff(u(x,t),x,x) = 0;
```

$$pde := \frac{\partial^2}{\partial t^2} u(x, t) - 4 \left(\frac{\partial^2}{\partial x^2} u(x, t) \right) = 0 \tag{3.1}$$

```
> conds := u(x,0) = exp(-x^2), D[2](u)(x,0) = 0;
```

$$conds := u(x, 0) = e^{-x^2}, D_2(u)(x, 0) = 0 \tag{3.2}$$

```
> pdsolve([pde, conds]);
```

$$u(x, t) = \frac{e^{-(2t-x)^2}}{2} + \frac{e^{-(2t+x)^2}}{2} \tag{3.3}$$

And an example with a source in the PDE:

```
> pde := diff(u(x,t),t,t) - 4*diff(u(x,t),x,x) = f(a);
```

$$pde := \frac{\partial^2}{\partial t^2} u(x, t) - 4 \left(\frac{\partial^2}{\partial x^2} u(x, t) \right) = f(a) \tag{3.4}$$

```
> conds := u(x,0) = 0, D[2](u)(x,0) = x^2;
```

$$conds := u(x, 0) = 0, D_2(u)(x, 0) = x^2 \tag{3.5}$$

```
> pdsolve([pde, conds]);
```

$$u(x,t) = \frac{f(a)\,t^2}{2} + \frac{4\,t^3}{3} + t\,x^2 \tag{3.6}$$

PDEtools commands for working with PDE

New PDEtools general-purpose commands and options for researching and solving PDEs have been implemented.

Convert a first-order PDE that contains the dependent variable explicitly into one that does not

The command, within the PDEtools package, is called ToMissingDependentVariable. It works by setting a new dependent variable which is a function of the independent variables of the problem, as well as of the old dependent variable. In the resulting PDE, the new dependent variable does not appear explicitly. For example, consider the following non-linear first order PDE which was not solved by pdsolve previously:

> $pde\_with\_m := y\cdot\left(diff(m(x,y),x)^2 - diff(m(x,y),y)^2\right) + m(x,y)\cdot diff\left(m(x,y),y\right);$

$$pde\_with\_m := y\left(m_x^{\,2} - m_y^{\,2}\right) + m(x,y)\,m_y \tag{4.1.1}$$

Its solution is now found by pdsolve by first converting it into a PDE without a dependent variable. For this, it uses the command ToMissingDependentVariable:

> $pde\_missing\_m := PDEtools\text{:-}ToMissingDependentVariable(pde\_with\_m, m(x,y), v);$

$$pde\_missing\_m := v_x^{\,2}\,y - m\,v_y\,v_m - v_y^{\,2}\,y,\ v(x,y,m) \tag{4.1.2}$$

That is, in order to solve pde_with_m, pdsolve creates and solves the above pde_missing_m, using a new function v(x,y,m), and then changes variables back to m(x,y) to give the solution:

> $pdsolve(pde\_with\_m);$

$$\left[\, m(x,y) = \mathrm{e}^{\frac{1}{2\_C3}\left(\_C3\ln(y) - 2\_C2\,x + \sqrt{4\_C2^2y^2 + \_C3^2} - 2\_C1 - \_C3\,\mathrm{csgn}(\_C3)\ln\left(\frac{2\_C3\left(\mathrm{csgn}(\_C3)\sqrt{4\_C2^2y^2 + \_C3^2} + \_C3\right)}{y}\right)\right)}\ \&\mathrm{where}\right] \tag{4.1.3}$$

[(*There are no arbitrary functions*)]

New option *reverse*, for computing the family of PDEs to which corresponds a given characteristic

strip, in the command charstrip

Consider a PDE, its characteristic strip, and its solution. What if we could send just the characteristic strip back to PDEtools and ask for all the PDEs that correspond to it? Now we can and more often than not, it is a whole family of PDEs that correspond to any given characteristic strip. This allows users to search for families of PDEs whose solution may be found due to knowing the (solvable) characteristic strip of only one member of such a family.

In the example below, note the use of the option "simplifyusingpde = false"; this is necessary so that the characteristic strip does not get simplified using the given PDE, rendering the "reverse" option as nonfunctional.

Example: Here is a PDE for which we know the characteristic strip:

> $pde := x*(diff(f(x,y,z),z))\text{^}2-f(x,y,z)+y\text{^}2*(diff(f(x,y,z),y))=0;$

$$pde := x f_z^2 - f + y^2 f_y = 0 \qquad \textbf{(4.2.1)}$$

> $PDEtools:-charstrip(pde, f(x,y,z), simplifyusingpde = false);$

$$\left\{ (\_p_2)_{\_s} = -2\,y(\_s)\,\_p_2(\_s) + \_p_2(\_s),\ (\_p_3)_{\_s} = \_p_3(\_s), f_{\_s} = 2\,x(\_s)\,\_p_3(\_s)^2 + y(\_s)^2\,\_p_2(\_s), x_{\_s} = 0, y_{\_s} = y(\_s)^2, z_{\_s} = 2\,x(\_s)\,\_p_3(\_s) \right\} \text{ \&where } \{\_p_2 = f_y, \_p_3 = f_z\} \qquad \textbf{(4.2.2)}$$

And here is the family of PDEs that corresponds to that characteristic strip (note the arbitrary function _*F1*(x)):

> $PDEtools:-charstrip(\%, f(x,y,z), reverse);$

$$-f + y^2 f_y + x f_z^2 + \_F1(x) \qquad \textbf{(4.2.3)}$$