A Symbolic Approach to Boundary Problems for Linear Partial Differential Equations

Applications to the Completely Reducible Case of the Cauchy Problem with Constant Coefficients

Markus Rosenkranz Nalina Phisanbut

School of Mathematics, Statistics and Actuarial Science
University of Kent, Canterbury, United Kingdom

CASC'13

Berlin, September 2013

Overview

Structure of the Talk:

Overview

Structure of the Talk:

- Abstract setup

Overview

Structure of the Talk:

- Abstract setup
- Completely reducible PDEs

Overview

Structure of the Talk:

- Abstract setup
- Completely reducible PDEs
- Review of PIDO System

Overview

Structure of the Talk:

- Abstract setup
- Completely reducible PDEs
- Review of PIDO System
- Glimpse of OPIDO Implementation

Overview

Structure of the Talk:

- Abstract setup
- Completely reducible PDEs
- Review of PIDO System
- Glimpse of OPIDO Implementation

Restriction: Regular boundary problems

Overview

Structure of the Talk:

- Abstract setup
- Completely reducible PDEs
- Review of PIDO System
- Glimpse of OPIDO Implementation

Restriction: Regular boundary problems
(Singular boundary problems for ODEs $\rightarrow$ [Korporal2012])

Typical Example

Given a forcing function $f(t, x, y)$ and
initial data $f_1(x, y), f_2(x, y)$, find $u(t, x, y)$ such that:

$$\boxed{\begin{array}{l} u_{tt} - 4\,u_{tx} + 4\,u_{xx} - 9\,u_{yy} = f \\ u(0, x, y) = f_1(x, y), \quad u_t(0, x, y) = f_2(x, y) \end{array}}$$

Given a forcing function $f(t,x,y)$ and
initial data $f_1(x,y), f_2(x,y)$, find $u(t,x,y)$ such that:

$$\boxed{\begin{array}{l} u_{tt} - 4\,u_{tx} + 4\,u_{xx} - 9\,u_{yy} = f \\ u(0,x,y) = f_1(x,y), \quad u_t(0,x,y) = f_2(x,y) \end{array}}$$

How can we capture this algebraically,

Typical Example

Given a forcing function $f(t, x, y)$ and initial data $f_1(x, y), f_2(x, y)$, find $u(t, x, y)$ such that:

$$\boxed{\begin{array}{l} u_{tt} - 4\,u_{tx} + 4\,u_{xx} - 9\,u_{yy} = f \\ u(0, x, y) = f_1(x, y), \quad u_t(0, x, y) = f_2(x, y) \end{array}}$$

How can we capture this algebraically, abstractly?

Abstract Setup: Recap

Starting Point: [RegensburgerRosenkranz2009]

Abstract Setup: Recap

Starting Point: [RegensburgerRosenkranz2009]

Definition

A generic boundary problem is given by a pair $(T, \mathcal{B})$, where $T \colon \mathcal{F} \to \mathcal{G}$ is an epimorphism between vector spaces $\mathcal{F}, \mathcal{G}$ and $\mathcal{B} \leq \mathcal{F}^*$ is an orthogonally closed subspace of boundary conditions.

It is called regular if $\mathrm{Ker}(T) \dotplus \mathcal{B}^\perp = \mathcal{F}$

Abstract Setup: Recap

Starting Point: [RegensburgerRosenkranz2009]

Definition

A generic boundary problem is given by a pair $(T, \mathcal{B})$, where $T\colon \mathcal{F} \to \mathcal{G}$ is an epimorphism between vector spaces $\mathcal{F}, \mathcal{G}$ and $\mathcal{B} \leq \mathcal{F}^*$ is an orthogonally closed subspace of boundary conditions.

It is called regular if $\mathrm{Ker}(T) \dotplus \mathcal{B}^\perp = \mathcal{F}$

Definition and Proposition

Define the product of two boundary problems $(T, \mathcal{B})$ and $(\tilde{T}, \tilde{\mathcal{B}})$ by

$$(T, \mathcal{B})(\tilde{T}, \tilde{\mathcal{B}}) = (T\tilde{T}, \mathcal{B}\tilde{T} + \tilde{\mathcal{B}}).$$

Then $(T, \mathcal{B})(\tilde{T}, \tilde{\mathcal{B}})$ is regular if both factors are.

Three Specific Incarnations

Three Specific Incarnations

Fully Inhomogeneous Boundary Problem:

$$\boxed{\begin{aligned} Tu &= \text{Forcing function} \\ \beta(u) &= \text{Boundary data} \end{aligned}}$$

Full Solution Operator

$$F\colon (\text{Forcing function}, \text{Boundary data}) \mapsto u$$

Three Specific Incarnations

Fully Inhomogeneous Boundary Problem:

$$\boxed{\begin{aligned} Tu &= \text{Forcing function} \\ \beta(u) &= \text{Boundary data} \end{aligned}}$$

Full Solution Operator

$F\colon$ (Forcing function, Boundary data) $\mapsto u$

Semi-Inhomogeneous Boundary Problem:

$$\boxed{\begin{aligned} Tu &= \text{Forcing function} \\ \beta(u) &= 0 \end{aligned}}$$

Signal Operator

$G\colon$ Forcing function $\mapsto u$

Three Specific Incarnations

Fully Inhomogeneous Boundary Problem:

$$\boxed{\begin{aligned} Tu &= \text{Forcing function} \\ \beta(u) &= \text{Boundary data} \end{aligned}}$$

Full Solution Operator

$F\colon$ (Forcing function, Boundary data) $\mapsto u$

Semi-Inhomogeneous Boundary Problem:

$$\boxed{\begin{aligned} Tu &= \text{Forcing function} \\ \beta(u) &= 0 \end{aligned}}$$

Signal Operator

$G\colon$ Forcing function $\mapsto u$

Semi-Homogeneous Boundary Problem:

$$\boxed{\begin{aligned} Tu &= 0 \\ \beta(u) &= \text{Boundary data} \end{aligned}}$$

State Operator

$H\colon$ Boundary data $\mapsto u$

Three Specific Incarnations

Fully Inhomogeneous Boundary Problem:

$$\boxed{\begin{aligned} Tu &= \text{Forcing function} \\ \beta(u) &= \text{Boundary data} \end{aligned}}$$

Full Solution Operator

$F\colon (\text{Forcing function}, \text{Boundary data}) \mapsto u$

Semi-Inhomogeneous Boundary Problem:

$$\boxed{\begin{aligned} Tu &= \text{Forcing function} \\ \beta(u) &= 0 \end{aligned}}$$

Signal Operator

$G\colon \text{Forcing function} \mapsto u$

Semi-Homogeneous Boundary Problem:

$$\boxed{\begin{aligned} Tu &= 0 \\ \beta(u) &= \text{Boundary data} \end{aligned}}$$

State Operator

$H\colon \text{Boundary data} \mapsto u$

Fully Homogeneous Boundary Problem:

$$\boxed{\begin{aligned} Tu &= 0 \\ \beta(u) &= 0 \end{aligned}}$$

Three Specific Incarnations

Fully Inhomogeneous Boundary Problem:

$$\boxed{\begin{aligned} Tu &= \text{Forcing function} \\ \beta(u) &= \text{Boundary data} \end{aligned}}$$

Full Solution Operator

F: (Forcing function, Boundary data) $\mapsto u$

Semi-Inhomogeneous Boundary Problem:

$$\boxed{\begin{aligned} Tu &= \text{Forcing function} \\ \beta(u) &= 0 \end{aligned}}$$

Signal Operator

G: Forcing function $\mapsto u$

Semi-Homogeneous Boundary Problem:

$$\boxed{\begin{aligned} Tu &= 0 \\ \beta(u) &= \text{Boundary data} \end{aligned}}$$

State Operator

H: Boundary data $\mapsto u$

Fully Homogeneous Boundary Problem:

$$\boxed{\begin{aligned} Tu &= 0 \\ \beta(u) &= 0 \end{aligned}}$$

Trivial: $u = 0$

Abstract Setup: What is "Boundary Data"?

$$\begin{cases} \beta_1(u) = 0 \\ \quad\vdots \\ \beta_n(u) = 0 \end{cases} \xrightarrow{\text{“basis-free”}} \mathcal{B} = [\beta_1, \ldots, \beta_n] \leq \mathcal{F}^*$$

Abstract Setup: What is "Boundary Data"?

$$\left\{\begin{array}{c} \beta_1(u) = c_1 \\ \vdots \\ \beta_n(u) = c_n \end{array}\right. \xrightarrow{\text{"basis-free"}} \quad \mathcal{B} = [\beta_1, \dots, \beta_n] \leq \mathcal{F}^*$$

???

Abstract Setup: What is “Boundary Data”?

$$\left\{ \begin{array}{c} \beta_1(u) = c_1 \\ \vdots \\ \beta_n(u) = c_n \end{array} \right. \xrightarrow{\text{“basis-free”}} \begin{array}{c} \mathcal{B} = [\beta_1, \dots, \beta_n] \leq \mathcal{F}^* \\ ??? \end{array}$$

Note: Inhomogeneous boundary conditions trivial for ODEs only!

Abstract Setup: What is “Boundary Data”?

$$\begin{cases} \beta_1(u) = c_1 \\ \quad\vdots \\ \beta_n(u) = c_n \end{cases} \xrightarrow{\text{“basis-free”}} \qquad \mathcal{B} = [\beta_1, \dots, \beta_n] \leq \mathcal{F}^*$$

???

Note: Inhomogeneous boundary conditions trivial for ODEs only!

Question: How to represent $(c_1, \dots, c_n)$ in a “basis-free” manner?

Abstract Setup: What is "Boundary Data"?

$$\left\{ \begin{array}{c} \beta_1(u) = c_1 \\ \vdots \\ \beta_n(u) = c_n \end{array} \right. \qquad \xrightarrow{\text{“basis-free”}} \qquad \begin{array}{c} \mathcal{B} = [\beta_1, \dots, \beta_n] \le \mathcal{F}^* \\ ??? \end{array}$$

Note: Inhomogeneous boundary conditions trivial for ODEs only!

Question: How to represent $(c_1, \dots, c_n)$ in a "basis-free" manner?

Answer: $(c_1, \dots, c_n) \in \mathcal{B}^*$ **!**

Abstract Setup: What is "Boundary Data"?

$$\left\{\begin{array}{c} \beta_1(u) = c_1 \\ \vdots \\ \beta_n(u) = c_n \end{array}\right. \quad \xrightarrow{\text{"basis-free"}} \quad \begin{array}{c} \mathcal{B} = [\beta_1, \dots, \beta_n] \leq \mathcal{F}^* \\ ??? \end{array}$$

Note: Inhomogeneous boundary conditions trivial for ODEs only!

Question: How to represent $(c_1, \dots, c_n)$ in a "basis-free" manner?

Answer: $(c_1, \dots, c_n) \in \mathcal{B}^*$ **!**

In this special case:

$$(\forall \beta \in \mathcal{B}) \quad (c_1, \dots, c_n)(\beta) = c_1 b_1 + \cdots + c_n b_n,$$
$$\text{where } \beta = b_1 \beta_1 + \cdots + b_n \beta_n.$$

Abstract Setup: What is "Boundary Data"?

$$\begin{cases} \beta_1(u) = c_1 \\ \quad\vdots \\ \beta_n(u) = c_n \end{cases} \xrightarrow{\text{"basis-free"}} \qquad \mathcal{B} = [\beta_1, \dots, \beta_n] \leq \mathcal{F}^*$$

???

Note: Inhomogeneous boundary conditions trivial for ODEs only!

Question: How to represent $(c_1, \dots, c_n)$ in a "basis-free" manner?

Answer: $(c_1, \dots, c_n) \in \mathcal{B}^*$ **!**

In this special case:

$$(\forall \beta \in \mathcal{B}) \quad (c_1, \dots, c_n)(\beta) = c_1 b_1 + \dots + c_n b_n,$$
$$\text{where } \beta = b_1\beta_1 + \dots + b_n\beta_n.$$

Generalize to PDEs.

Abstract Setup: Trace Map

Definition

Let $\mathcal{F}, \mathcal{G}$ be K-vector spaces and $\mathcal{B} \leq \mathcal{F}^*$ an orthogonally closed subspace of boundary conditions. The trace map $\operatorname{trc}\colon \mathcal{F} \to \mathcal{B}^*$ sends $f \in \mathcal{F}$ to the functional $\beta \mapsto \beta(f)$ with $\mathcal{B}' := \operatorname{Im}(\operatorname{trc}) \leq \mathcal{B}^*$.

Abstract Setup: Trace Map

Definition

Let $\mathcal{F}, \mathcal{G}$ be K-vector spaces and $\mathcal{B} \leq \mathcal{F}^*$ an orthogonally closed subspace of boundary conditions. The trace map $\operatorname{trc}\colon \mathcal{F} \to \mathcal{B}^*$ sends $f \in \mathcal{F}$ to the functional $\beta \mapsto \beta(f)$ with $\mathcal{B}' := \operatorname{Im}(\operatorname{trc}) \leq \mathcal{B}^*$.

"Boundary Data" $:=$ Elements of $\mathcal{B}'$

Abstract Setup: Trace Map

Definition

Let $\mathcal{F}, \mathcal{G}$ be K-vector spaces and $\mathcal{B} \le \mathcal{F}^*$ an orthogonally closed subspace of boundary conditions. The trace map $\operatorname{trc}\colon \mathcal{F} \to \mathcal{B}^*$ sends $f \in \mathcal{F}$ to the functional $\beta \mapsto \beta(f)$ with $\mathcal{B}' := \operatorname{Im}(\operatorname{trc}) \le \mathcal{B}^*$.

"Boundary Data" $:=$ Elements of $\mathcal{B}'$

$$\begin{array}{ccc} \text{Boundary Data} & \xrightarrow{\text{Boundary Basis } (\beta_i)_{i \in I}} & \text{Boundary Values} \\ B \in \mathcal{B}' & & \bar{B} = B(\beta_i)_{i \in I} \in K^I \end{array}$$

Abstract Setup: Trace Map

Definition

Let $\mathcal{F}, \mathcal{G}$ be K-vector spaces and $\mathcal{B} \leq \mathcal{F}^*$ an orthogonally closed subspace of boundary conditions. The trace map $\operatorname{trc}\colon \mathcal{F} \to \mathcal{B}^*$ sends $f \in \mathcal{F}$ to the functional $\beta \mapsto \beta(f)$ with $\mathcal{B}' := \operatorname{Im}(\operatorname{trc}) \leq \mathcal{B}^*$.

"Boundary Data" $:=$ Elements of $\mathcal{B}'$

Boundary Data	$\xrightarrow{\text{Boundary Basis } (\beta_i)_{i \in I}}$	Boundary Values
$B \in \mathcal{B}'$		$\bar{B} = B(\beta_i)_{i \in I} \in K^I$
basis-free		basis-dependent

Abstract Setup: Trace Map

Definition

Let $\mathcal{F}, \mathcal{G}$ be K-vector spaces and $\mathcal{B} \leq \mathcal{F}^*$ an orthogonally closed subspace of boundary conditions. The trace map $\mathrm{trc}\colon \mathcal{F} \to \mathcal{B}^*$ sends $f \in \mathcal{F}$ to the functional $\beta \mapsto \beta(f)$ with $\mathcal{B}' := \mathrm{Im}(\mathrm{trc}) \leq \mathcal{B}^*$.

"Boundary Data" := Elements of $\mathcal{B}'$

$$\begin{array}{ccc} \text{Boundary Data} & & \text{Boundary Values} \\ B \in \mathcal{B}' & \xrightarrow{\text{Boundary Basis } (\beta_i)_{i \in I}} & \bar{B} = B(\beta_i)_{i \in I} \in K^I \\ \text{basis-free} & & \text{basis-dependent} \end{array}$$

Lemma

Let $\mathcal{B} \leq \mathcal{F}^*$ be a boundary space with boundary basis $(\beta_i \mid \in I)$. If for any $B, \tilde{B} \in \mathcal{B}'$ one has $B(\beta_i)_{i \in I} = \tilde{B}(\beta_i)_{i \in I}$ then also $B = \tilde{B}$. In particular, for any $f \in \mathcal{F}$, the trace $f^* := \mathrm{trc}\colon \mathcal{F} \to \mathcal{B}^*$ depends only on the boundary values $f(\beta_i)_{i \in I}$.

Abstract Setup: Right Inverse and Interpolator

Definition

We write $T^\diamond$ for any right inverse of T. An interpolator for $\mathcal{B}$ is any right inverse $\mathcal{B}^\diamond : \mathcal{B}' \to \mathcal{F}$ of the trace map $\operatorname{trc} : \mathcal{F} \to \mathcal{B}^*$.

Abstract Setup: Right Inverse and Interpolator

Definition

We write $T^\diamond$ for any right inverse of T. An interpolator for $\mathcal{B}$ is any right inverse $\mathcal{B}^\diamond \colon \mathcal{B}' \to \mathcal{F}$ of the trace map $\operatorname{trc} \colon \mathcal{F} \to \mathcal{B}^*$.

Computation of Green's Operator decomposes into differential equation/boundary conditions.

Proposition

Let $(T, \mathcal{B})$ be regular boundary problem. Then $G = (1 - P)\, T^\diamond$ and $H = P\mathcal{B}^\diamond$, hence $F = (1 - P)\, T^\diamond \oplus P\mathcal{B}^\diamond$. Here $P \colon \mathcal{F} \to \mathcal{F}$ is the projector determined by $\operatorname{Im}(P) = \operatorname{Ker}(T)$ and $\operatorname{Ker}(P) = \mathcal{B}^\perp$.

How Do We Get the Kernel Projector?

How Do We Get the Kernel Projector?

In the ODE case:

Proposition

For a regular boundary problem $(T, \mathcal{B})$ with $\mathrm{Ker}(T) = [u_1, \dots, u_n]$ and $\mathcal{B} = [\beta_1, \dots, \beta_n]$, the kernel projector is given by $P = (u_1, \dots, u_n)\ \beta(u)^{-1}(\beta_1, \dots, \beta_n)^\intercal$.

How Do We Get the Kernel Projector?

In the ODE case:

Proposition

For a regular boundary problem $(T, \mathcal{B})$ with $\operatorname{Ker}(T) = [u_1, \dots, u_n]$ and $\mathcal{B} = [\beta_1, \dots, \beta_n]$, the kernel projector is given by $P = (u_1, \dots, u_n)\ \beta(u)^{-1}(\beta_1, \dots, \beta_n)^{\mathsf{T}}$.

Problem: Row elimination on infinitely many rows?!

How Do We Get the Kernel Projector?

In the ODE case:

Proposition

For a regular boundary problem $(T, \mathcal{B})$ with $\operatorname{Ker}(T) = [u_1, \ldots, u_n]$ and $\mathcal{B} = [\beta_1, \ldots, \beta_n]$, the kernel projector is given by $P = (u_1, \ldots, u_n)\ \beta(u)^{-1} (\beta_1, \ldots, \beta_n)^\intercal$.

Problem: Row elimination on infinitely many rows?!
Need more intuitive description of P.

How Do We Get the Kernel Projector?

In the ODE case:

Proposition

For a regular boundary problem $(T, \mathcal{B})$ with $\mathrm{Ker}(T) = [u_1, \dots, u_n]$ and $\mathcal{B} = [\beta_1, \dots, \beta_n]$, the kernel projector is given by $P = (u_1, \dots, u_n)\, \beta(u)^{-1}(\beta_1, \dots, \beta_n)^\intercal$.

Problem: Row elimination on infinitely many rows?!
Need more intuitive description of P.

Proposition

Let $(T, \mathcal{B})$ be a regular boundary problem with $E \colon \mathrm{Ker}(T) \to \mathcal{B}'$ being the restricted trace map. Then E is bijective with the state operator H as its inverse, and $P = H \circ \mathrm{trc}$ is the projector with $\mathrm{Im}(P) = \mathrm{Ker}(T)$ and $\mathrm{Ker}(P) = \mathcal{B}^\perp$.

Back to Real: Cauchy Problem for Analytic Functions

Theorem (Global Cauchy-Kovalevskaya) [Knapp2005]

Let $T \in \mathbb{C}[D_t, D_1, \ldots, D_n]$ be a differential operator in Cauchy-Kovalevskaya form with respect to t,
meaning $T = D_t^m + \tilde{T}$ with $\deg(\tilde{T}, t) < m$ and $\deg(\tilde{T}) \leq m$.
Then the Cauchy problem

$$\left.\begin{array}{l} Tu = 0 \\ D_t^{i-1} u(0, x_1, \ldots, x_n) = f_i(x_1, \ldots, x_n) \text{ for } i = 1, \ldots, m \end{array}\right\} \quad (1)$$

has a unique solution $u \in C^\omega(\mathbb{R}^{n+1})$ for
given $(f_1, \ldots, f_m) \in C^\omega(\mathbb{R}^n)^m$.

Back to Real: Cauchy Problem for Analytic Functions

Theorem (Global Cauchy-Kovalevskaya) [Knapp2005]

Let $T \in \mathbb{C}[D_t, D_1, \ldots, D_n]$ be a differential operator in Cauchy-Kovalevskaya form with respect to t, meaning $T = D_t^m + \tilde{T}$ with $\deg(\tilde{T}, t) < m$ and $\deg(\tilde{T}) \leq m$. Then the Cauchy problem

$$\left.\begin{aligned} &Tu = 0 \\ &D_t^{i-1} u(0, x_1, \ldots, x_n) = f_i(x_1, \ldots, x_n) \text{ for } i = 1, \ldots, m \end{aligned}\right\} \quad (1)$$

has a unique solution $u \in C^\omega(\mathbb{R}^{n+1})$ for given $(f_1, \ldots, f_m) \in C^\omega(\mathbb{R}^n)^m$.

Note: Boundary problem is regular but may be ill-posed.

Composition of Boundary Problem

Since we restrict ourselves to completely reducible T:

Composition of Boundary Problem

Since we restrict ourselves to completely reducible T:

Proposition

Let $(T, \mathcal{B})$ and $(\tilde{T}, \tilde{\mathcal{B}})$ be regular problems with the signal operators G, $\tilde{G}$ and the state operators H, $\tilde{H}$. Then $(T, \mathcal{B})(\tilde{T}, \tilde{\mathcal{B}})$ has the signal operator $\tilde{G}G$ and the state operator $(\mathcal{B}\tilde{T} + \tilde{\mathcal{B}})' \to \mathcal{F}$ acting by $B + \tilde{B} \mapsto \tilde{G}H(B\tilde{T}^*) + \tilde{H}(\tilde{B})$.

Composition of Boundary Problem

Since we restrict ourselves to completely reducible T:

Proposition

Let $(T, \mathcal{B})$ and $(\tilde{T}, \tilde{\mathcal{B}})$ be regular problems with the signal operators G, $\tilde{G}$ and the state operators H, $\tilde{H}$. Then $(T, \mathcal{B})(\tilde{T}, \tilde{\mathcal{B}})$ has the signal operator $\tilde{G}G$ and the state operator $(\mathcal{B}\tilde{T} + \tilde{\mathcal{B}})' \to \mathcal{F}$ acting by $B + \tilde{B} \mapsto \tilde{G}H(B\tilde{T}^*) + \tilde{H}(\tilde{B})$.

Lemma

Let $T = a + a_0\partial_t + a_1\partial_1 + \cdots + a_n\partial_n \in \mathbb{C}[D]$ be a first-order operator with all $a_i \neq 0$. Then the Cauchy problem $Tu = 0$, $u(0, x_1, \ldots, x_n) = f(x_1, \ldots, x_n)$ has state operator $H(f) = e^{-at/a_0}\, Z^* \tilde{Z}_x^* f$ and signal operator $G = a_0^{-1}\, e^{at/a_0}\, Z^*\, A_t\, e^{-at/a_0}\, \tilde{Z}^*$, with $Z = Z(a_0, a_1, \ldots, a_n)$ where Z has inverse $\tilde{Z}$.

Partial Integro-Differential Operators (PIDOS)

Definition

The partial integro-differential operators are the complex algebra generated by the indeterminates below, modulo certain rewrite rules. Notation $\mathcal{F}[\partial_x, \partial_y, \int^x, \int^y]$.

Partial Integro-Differential Operators (PIDOS)

Definition

The partial integro-differential operators are the complex algebra generated by the indeterminates below, modulo certain rewrite rules. Notation $\mathcal{F}[\partial_x, \partial_y, \int^x, \int^y]$.

Name	Indeterminates	Range	Action on $u(x, y)$
Substitutions	$\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)^*$	$a,b,c,d \in \mathbb{C}$	$u(ax + by, cx + dy)$
Rotations	Q_α^*	$\alpha \in [0, 2\pi]$	$u(\gamma x - \sigma y, \sigma x + \gamma y)$
Multipliers	$e^{\lambda x} x^m$, $e^{\mu y} y^n$	$m,n \in \mathbb{N}^+$, $\lambda,\mu \in \mathbb{C}$	$e^{\lambda x} x^m\, u(x, y)$, $e^{\mu y} y^n\, u(x, y)$
Integrations	A_x, A_y	–	$\int_0^x u(\xi, y)\, d\xi, \int_0^y u(x, \eta)\, d\eta$
Derivations	D_x, D_y	–	$u_x(x, y)$, $u_y(x, y)$

Note: Still lacking confluence proof for rewrite system!

One-Dimensional Substitution Rule:

$$A_x x^\mu \begin{pmatrix} a & b \\ 0 & d \end{pmatrix}^* = \begin{cases} \frac{1}{a^{\mu+1} d^\mu} \left(1 - L_x\right) \begin{pmatrix} a & b \\ 0 & d \end{pmatrix}^* A_x \left(dx - by\right)^\mu & \text{for } ad \neq 0 \\ \frac{1}{a^{\mu+1}} \left(1 - L_x\right) \begin{pmatrix} a & b \\ 0 & 1 \end{pmatrix}^* A_x \left(x - by\right)^\mu L_y & \text{for } ab \neq 0,\ d = 0 \\ \frac{1}{a^{\mu+1}} \begin{pmatrix} a & 0 \\ 0 & 0 \end{pmatrix}^* A_x\, x^\mu & \text{for } a \neq 0,\ b = d = 0 \\ \frac{1}{\mu+1}\, x^{\mu+1} \begin{pmatrix} 0 & b \\ 0 & d \end{pmatrix}^* & \text{for } a = 0 \end{cases}$$

Here $L_x \equiv \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}^*$, $L_y \equiv \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}^*$ are the evaluations $x \mapsto 0$, $y \mapsto 0$.

PIDOS: Some Rewrite Rules

One-Dimensional Substitution Rule:

$$A_x x^\mu \left(\begin{smallmatrix} a & b \\ 0 & d \end{smallmatrix}\right)^* = \begin{cases} \frac{1}{a^{\mu+1} d^\mu} \, (1 - L_x) \left(\begin{smallmatrix} a & b \\ 0 & d \end{smallmatrix}\right)^* A_x \, (dx - by)^\mu & \text{for } ad \neq 0 \\ \frac{1}{a^{\mu+1}} \, (1 - L_x) \left(\begin{smallmatrix} a & b \\ 0 & 1 \end{smallmatrix}\right)^* A_x \, (x - by)^\mu L_y & \text{for } ab \neq 0, \ d = 0 \\ \frac{1}{a^{\mu+1}} \left(\begin{smallmatrix} a & 0 \\ 0 & 0 \end{smallmatrix}\right)^* A_x \, x^\mu & \text{for } a \neq 0, \ b = d = 0 \\ \frac{1}{\mu+1} \, x^{\mu+1} \left(\begin{smallmatrix} 0 & b \\ 0 & d \end{smallmatrix}\right)^* & \text{for } a = 0 \end{cases}$$

Here $L_x \equiv \left(\begin{smallmatrix} 0 & 0 \\ 0 & 1 \end{smallmatrix}\right)^*$, $L_y \equiv \left(\begin{smallmatrix} 1 & 0 \\ 0 & 0 \end{smallmatrix}\right)^*$ are the evaluations $x \mapsto 0$, $y \mapsto 0$.

Two-Dimensional Substitution Rule:

$$A_x Q_\alpha^* A_x Q_{\tilde\alpha}^* = \tfrac{1}{\sigma}(1 - L_x) \left[(\sigma\tilde\sigma - \gamma\tilde\gamma) \, A_x Q_{\alpha+\tilde\alpha}^* + \tilde\sigma \left(\begin{smallmatrix} -\sigma & -\gamma \\ 0 & 0 \end{smallmatrix}\right)^* A_x Q_{\tilde\alpha - \frac{\pi}{2}}^* + \tilde\gamma \, Q_\alpha^* A_x Q_{\tilde\alpha}^* \right] A_y$$

$$A_x Q_\alpha^* A_y Q_{\tilde\alpha}^* = \tfrac{1}{\gamma}(1 - L_x) \left[(\gamma\tilde\gamma - \sigma\tilde\sigma) \, A_x Q_{\alpha+\tilde\alpha}^* - \tilde\gamma \left(\begin{smallmatrix} \gamma & -\sigma \\ 0 & 0 \end{smallmatrix}\right)^* A_x Q_{\tilde\alpha}^* + \tilde\sigma \, Q_{\alpha - \frac{\pi}{2}}^* A_x Q_{\tilde\alpha + \frac{\pi}{2}}^* \right] A_y$$

Back to the Initial Example

$$\boxed{\begin{array}{l} u_{tt} - 4\,u_{tx} + 4\,u_{xx} - 9\,u_{yy} = f, \\ u(0,x,y) = f_1(x,y), \quad u_t(0,x,y) = f_2(x,y) \end{array}}$$

Back to the Initial Example

$$\boxed{\begin{array}{l} u_{tt} - 4\,u_{tx} + 4\,u_{xx} - 9\,u_{yy} = f, \\ u(0,x,y) = f_1(x,y), \quad u_t(0,x,y) = f_2(x,y) \end{array}}$$

The signal and state operators:

$$Gf\,(t,x,y) = \int_0^t \int_0^\sigma f(\tau, x + 2t - 2\tau, y - 3t - 3\tau + 6\sigma)\,d\tau\,d\sigma.$$

$$H(f_1,f_2) = f_1(x+2t, y-3t) + \int_0^t (f_2 - 2\,D_x f_1 + 3\,D_y f_1)(x+2t, y-3t+6\tau)\,d\tau$$

Back to the Initial Example

$$\boxed{\begin{array}{l} u_{tt} - 4\,u_{tx} + 4\,u_{xx} - 9\,u_{yy} = f, \\ u(0,x,y) = f_1(x,y), \quad u_t(0,x,y) = f_2(x,y) \end{array}}$$

The signal and state operators:

$$Gf\,(t,x,y) = \int_0^t \int_0^\sigma f(\tau, x+2t-2\tau, y-3t-3\tau+6\sigma)\,d\tau\,d\sigma.$$

$$H(f_1,f_2) = f_1(x{+}2t, y{-}3t) + \int_0^t (f_2 - 2\,D_x f_1 + 3\,D_y f_1)(x{+}2t, y{-}3t{+}6\tau)\,d\tau$$

Factor problems:

$$\boxed{\begin{array}{l} u_t - 2\,u_x \pm 3\,u_y = f, \\ u(0,x,y) = f^{\pm}(x,y). \end{array}}$$

$$H^{\pm} f^{\pm}\,(t,x,y) = f^{\pm}(x+2t, y \mp 3t)$$

$$G^{\pm} f\,(t,x,y) = \int_0^t f(\tau, x+2t-2\tau, y \mp 3t \pm 3\tau)\,d\tau$$

OPIDO Package

OPIDO = Ordinary and Partial Integro-Differential Operators

OPIDO Package

OPIDO = Ordinary and Partial Integro-Differential Operators

- Under development.

OPIDO Package

OPIDO = Ordinary and Partial Integro-Differential Operators

- Under development.
- Written in FUNPRO language.

OPIDO Package

OPIDO = Ordinary and Partial Integro-Differential Operators

- Under development.
- Written in FUNPRO language.

Mathematical domains:

OPIDO Package

OPIDO = Ordinary and Partial Integro-Differential Operators

- Under development.
- Written in FUNPRO language.

Mathematical domains:

- Every domain is represented by unique tag $\mathcal{D}$.

OPIDO Package

OPIDO = Ordinary and Partial Integro-Differential Operators

- Under development.
- Written in FUNPRO language.

Mathematical domains:

- Every domain is represented by unique tag $\mathcal{D}$.
- Every domain is generated with signature.

OPIDO Package

OPIDO = Ordinary and Partial Integro-Differential Operators

- Under development.
- Written in FUNPRO language.

Mathematical domains:

- Every domain is represented by unique tag $\mathcal{D}$.
- Every domain is generated with signature.
- Every domain has various operations e.g. $+_{\mathcal{D}}$.

OPIDO Package

OPIDO = Ordinary and Partial Integro-Differential Operators

- Under development.
- Written in FUNPRO language.

Mathematical domains:

- Every domain is represented by unique tag $\mathcal{D}$.
- Every domain is generated with signature.
- Every domain has various operations e.g. $+_{\mathcal{D}}$.
- A domain can be created from another domain.

OPIDO Package

OPIDO $=$ Ordinary and Partial Integro-Differential Operators

- Under development.
- Written in FUNPRO language.

Mathematical domains:

- Every domain is represented by unique tag $\mathcal{D}$.
- Every domain is generated with signature.
- Every domain has various operations e.g. $+_{\mathcal{D}}$.
- A domain can be created from another domain.

Parsing and formatting:

OPIDO Package

OPIDO = Ordinary and Partial Integro-Differential Operators

- Under development.
- Written in FUNPRO language.

Mathematical domains:

- Every domain is represented by unique tag $\mathcal{D}$.
- Every domain is generated with signature.
- Every domain has various operations e.g. $+_{\mathcal{D}}$.
- A domain can be created from another domain.

Parsing and formatting:

- Automated generation of special parsing and formatting per-domain basis.

OPIDO Package

OPIDO = Ordinary and Partial Integro-Differential Operators

- Under development.
- Written in FUNPRO language.

Mathematical domains:

- Every domain is represented by unique tag $\mathcal{D}$.
- Every domain is generated with signature.
- Every domain has various operations e.g. $+_{\mathcal{D}}$.
- A domain can be created from another domain.

Parsing and formatting:

- Automated generation of special parsing and formatting per-domain basis.
- Allow us to write integro-differential operators in a notation close to that on paper.

OPIDO versus GenPolyDom

Previous Implementation `GenPolyDom`	Current Implementation `OPIDO`

OPIDO versus GenPolyDom

Previous Implementation `GenPolyDom`	Current Implementation `OPIDO`
Written in the THEOREMA language	Standalone Mathematica package

OPIDO versus GenPolyDom

Previous Implementation `GenPolyDom`	Current Implementation `OPIDO`
Written in the THEOREMA language	Standalone Mathematica package
Uses underscripts $a \underset{\mathcal{D}}{+} b$	Uses subscripts: $a +_{\mathcal{D}} b$

OPIDO versus GenPolyDom

Previous Implementation `GenPolyDom`	Current Implementation `OPIDO`
Written in the THEOREMA language	Standalone Mathematica package
Uses underscripts $a \underset{\mathcal{D}}{+} b$	Uses subscripts: $a +_{\mathcal{D}} b$
Uses currying: $a \underset{\mathcal{D}}{+} b \rightsquigarrow \mathcal{D}[+][a, b]$	Uses tagging: $a +_{\mathcal{D}} b \rightsquigarrow \mathtt{DomOp}[\mathcal{D}, +, \mathtt{a}, \mathtt{b}]$

Examples

See Mathematica notebook.

Conclusion and Outlook

Existing setup:

Conclusion and Outlook

Existing setup:

- Abstract setup suitable for LPDE boundary problems.

Conclusion and Outlook

Existing setup:

- Abstract setup suitable for LPDE boundary problems.
- Algebraic representation of operators via PIDOS.

Conclusion and Outlook

Existing setup:

- Abstract setup suitable for LPDE boundary problems.
- Algebraic representation of operators via PIDOS.
- Implementation in progress.

Conclusion and Outlook

Existing setup:

- Abstract setup suitable for LPDE boundary problems.
- Algebraic representation of operators via PIDOS.
- Implementation in progress.

Future steps:

Conclusion and Outlook

Existing setup:

- Abstract setup suitable for LPDE boundary problems.
- Algebraic representation of operators via PIDOS.
- Implementation in progress.

Future steps:

- Finish completely reducible case: Stay within PIDOS.

Conclusion and Outlook

Existing setup:

- Abstract setup suitable for LPDE boundary problems.
- Algebraic representation of operators via PIDOS.
- Implementation in progress.

Future steps:

- Finish completely reducible case: Stay within PIDOS.
- Hyperbolic IVPs: Probably need Fourier transform.

Conclusion and Outlook

Existing setup:

- Abstract setup suitable for LPDE boundary problems.
- Algebraic representation of operators via PIDOS.
- Implementation in progress.

Future steps:

- Finish completely reducible case: Stay within PIDOS.
- Hyperbolic IVPs: Probably need Fourier transform.
- General case: Use Ehrenpreis-Palamodov Theorem?

Conclusion and Outlook

Existing setup:

- Abstract setup suitable for LPDE boundary problems.
- Algebraic representation of operators via PIDOS.
- Implementation in progress.

Future steps:

- Finish completely reducible case: Stay within PIDOS.
- Hyperbolic IVPs: Probably need Fourier transform.
- General case: Use Ehrenpreis-Palamodov Theorem?

Thank you!

References

Sönke Hansen.
On the "fundamental principle" of L. Ehrenpreis.
In *Partial differential equations (Warsaw, 1978)*, volume 10 of *Banach Center Publ.*. PWN, Warsaw, 1983.

Anthony W. Knapp.
Advanced real analysis.
Cornerstones. Birkhäuser Boston Inc., Boston, MA, 2005.

Anja Korporal.
Symbolic Methods for Generalized Green's Operators and Boundary Problems.
PhD thesis, Research Institute for Symbolic Computation, Johannes Kepler University, Linz, Austria, 2012.

Markus Rosenkranz and Nalina Phisanbut.
A symbolic approach to boundary problems for linear partial differential equations.
In *CASC 2013*. To appear.

Georg Regensburger and Markus Rosenkranz.
An algebraic foundation for factoring linear boundary problems.
Ann. Mat. Pura Appl. (4).

Markus Rosenkranz and Georg Regensburger.
Solving and factoring boundary problems for linear ordinary differential equations in differential algebras.
Journal of Symbolic Computation, 43(8):515–544, 2008.

Markus Rosenkranz, Georg Regensburger, Loredana Tec, and Bruno Buchberger.
Symbolic analysis of boundary problems: From rewriting to parametrized Gröbner bases.
In Ulrich Langer and Peter Paule, editors, *Numerical and Symbolic Scientific Computing: Progress and Prospects*. Springer, 2012.

Loredana Tec.
A Symbolic Framework for General Polynomial Domains in Theorema: Applications to Boundary Problems.
PhD thesis, Research Institute for Symbolic Computation, Johannes Kepler University, Linz, Austria, 2011.