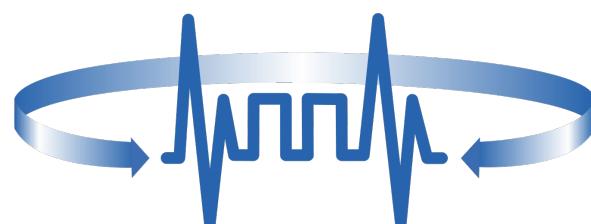


Сочетание различных методов дискретизации композиции неизвестных функций в численных методах сбалансированной идентификации математических моделей

Владимир Волошинов, Александр Соколов
<http://bit.ly/VoloshinovGScholar>

Центр распределённых вычислений
Институт проблем передачи информации (им. Харкевича) РАН,



Исследование выполнено за счет гранта Российского научного фонда
№ 25-11-00349, <https://rscf.ru/project/25-11-00349/>

SvF-технология сбалансированной идентификации

SvF (Simplicity vs Fitting) – давно развивающаяся технология построения математических моделей на основе экспериментальных данных. Была предложена Соколовым А.В. более 9 лет назад.

Основные публикации по принципам SvF

- Sokolov A., Voloshinov V. **Model Selection by Balanced Identification: the Interplay of Optimization and Distributed Computing** // Open Computer Science, **2020**
- Соколов А.В., Волошинов В.В. **Выбор математической модели: баланс между сложностью и близостью к измерениям** // International Journal of Open Information Technologies, **2018**

Успешно применялась в биологии, популяционной динамике, физиологии растений, физике плазмы, метеорологии, экологии ...

Свободна доступна в исходных кодах <https://github.com/distcomp/SvF> (список более 20 публикаций в разделе References, <https://github.com/distcomp/SvF#references>)

Вычислительная схема SvF основана на решении задач 2-х уровневой оптимизации специального вида, где на нижнем уровне нужно решать набор задач, сходных с обратными задачами метода регуляризации Тихонова. В докладе рассматривается один из типов таких задач.

Обратная задача для дифференциальных уравнений

«Демонстрационные» задачи определения правой части автономного ДУ 1, 2-го порядка по неточным данным о траектории

$$\left. \begin{array}{l} \left\{ (\tilde{x}_d, t_d) : d=1:N_d \right\}, \tilde{x}_d = x(t_d) \pm \text{err} \\ \dot{x} = F(x(t)) \mid \ddot{x} = F(x(t)), \quad t \in [t_L, t_U] \end{array} \right\} \Rightarrow \mathbf{x}(t), \mathbf{F}(\mathbf{x}) - ??$$

Опуская важные детали SvF-технологии, она требует многократного решения задач оптимизации следующего вида (здесь, фактически, - вариационных задач)

data fitting

simplicity (гладкость., штраф за резкие колебания)

$$\frac{1}{D} \sum_{d=1:D} (\tilde{x}_d - x(t_d))^2 + \alpha \int_{x_L}^{x_U} (F''(x))^2 dx \rightarrow \min_{\mathbf{x}(\cdot), \mathbf{F}(\cdot)},$$

$$\dot{x} = F(x(t)) \mid \ddot{x} = F(x(t)), \quad t \in [t_L, t_U],$$

$$x(\cdot) \in C^2 [t_L, t_U], F(\cdot) \in C^2 [x_L, x_U].$$

При замене вариационной задачи на её аналог в форме **конечномерной задачи математического программирования** используется дискретизация ДУ, интегралов и **суперпозиции** (!) неизвестных функций $F(x)$, $x(t)$

Обратная задача для ОДУ (неавтономного)

Определение правой части ОДУ 1, 2-го порядка по неточным данным о траектории

$$\left. \begin{array}{l} \dot{x} = F(x(t), t) \mid \ddot{x} = F(x(t), t), \quad t \in [t_L, t_U] \\ \{(\tilde{x}_d, t_d) : d=1:N_d\}, \tilde{x}_d = x(t_d) \pm \text{err} \end{array} \right\} \Rightarrow \mathbf{x}(t), \mathbf{F}(\mathbf{x}, t) - ??$$

SvF-технология, требует многократного решения задач оптимизации следующего вида (фактически, - вариационных задач)

$$\left\{ \begin{array}{l} \text{data fitting} \quad \text{simplicity (гладкость., штраф за резкие колебания)} \\ \frac{1}{D} \sum_{d=1:D} (\tilde{x}_d - x(t_d))^2 + R(\alpha_t, \alpha_x, F(\cdot, \cdot)) \rightarrow \min_{x(\cdot), F(\cdot, \cdot)}, \\ \dot{x} = F(x(t), t) \mid \ddot{x} = F(x(t), t), \quad t \in [t_L, t_U], \\ x(\cdot) \in C^2[t_L, t_U], F(\cdot, \cdot) \in C^2[x_L, x_U] \times [t_L, t_U]. \\ R(\alpha_x, \alpha_t, F(\cdot, \cdot)) = \\ \int_{t_L}^{t_U} \int_{x_L}^{x_U} \left(\alpha_t^2 (F''_{tt}(t, x))^2 + 2\alpha_t \alpha_x (F''_{tx}(t, x))^2 + \alpha_x^2 (F''_{xx}(t, x))^2 \right) dt dx \end{array} \right.$$

Для получения **конечномерной задачи математического программирования** нужна дискретизация ДУ, интегралов и **суперпозиции** функций $F(x, t), x(t)$

Дискретизация автономного ДУ: полином + сетка

Ранее в SvF: «внешняя» функция заменялась многочленом с неизвестными коэффициентами, а «внутренняя» — равномерной сеткой значений:

$$F(x) \approx \mathcal{P}(\mathbf{c}, x) = \sum_{p=0}^P \mathbf{c}_p \cdot x^p$$

$$x(t) \approx \{(\mathbf{x}(\mathbf{t}_k), t_k) : k=0:N_t\}, \\ t_k = t_{\mathbf{L}} + k \cdot \Delta t, \Delta t = \frac{t_{N_t} - t_{\mathbf{L}}}{N_t},$$

$$x_i = x_{\mathbf{L}} + i \cdot \Delta x, \Delta x = \frac{x_{N_x} - x_{\mathbf{L}}}{N_x}.$$

$$\left\{ \dot{x} = F(x(t)) \right\} \approx \left\{ \frac{x(t_k) - x(t_{k-1})}{\Delta t} = \sum_{p=0}^P c_p \cdot \left(\frac{x(t_k) + x(t_{k-1})}{2} \right)^p, k=1:N_t \right\}$$

$$\left\{ \ddot{x} = F(x(t)) \right\} \approx \left\{ \frac{x(t_{k+1}) - 2x(t_k) + x(t_{k-1})}{\Delta t^2} = \sum_{p=0}^P c_p \cdot x(t_k)^p, k=1:(N_t-1) \right\}$$

$$x(t_d) \approx \hat{x}_d = \frac{t_k - t_d}{\Delta t} x_{k-1} + \frac{t_d - t_{k-1}}{\Delta t} x_k, \quad t_d \in [t_{k-1}, t_k]$$

$$\frac{1}{D} \sum_{d=1:D} (\tilde{x}_d - \hat{x}_d)^2 + \alpha \sum \left(\frac{\mathcal{P}(x_{i+1}) - 2\mathcal{P}(x_i) + \mathcal{P}(x_{i-1})}{\Delta x^2} \right)^2 \Delta x \rightarrow \min_{\mathbf{x}(\mathbf{t}_k), \mathbf{c}_p} .$$

**Нелинейная задача математического программирования
с полиномами не малых (!) степеней**

Дискретизация автономного ДУ: ряд Фурье + сетка

Затем была добавлена возможность использовать еще «суммы» Фурье (с неизвестными коэффициентами и, возможно, периодом):

$$F(x) \approx \Phi(\mathbf{a}, \mathbf{b}, \boldsymbol{\omega}, x) = x(t) \approx \{(\mathbf{x}(t_k), t_k) : k=0:N_t\},$$

$$t_k = t_{\mathbf{L}} + k \cdot \Delta t, \Delta t = \frac{t_{N_t} - t_{\mathbf{L}}}{N_t},$$

$$x_i = x_{\mathbf{L}} + i \cdot \Delta x, \Delta x = \frac{x_{N_x} - x_{\mathbf{L}}}{N_x}.$$

$$\left\{ \dot{x} = F(x(t)) \right\} \approx \frac{x(t_k) - x(t_{k-1})}{\Delta t} = \Phi \left(\mathbf{a}, \mathbf{b}, \boldsymbol{\omega}, \frac{x(t_k) + x(t_{k-1})}{2} \right) (k=1:N_t)$$

$$x(t_d) \approx \hat{x}_d = \frac{t_k - t_d}{\Delta t} x(t_{k-1}) + \frac{t_d - t_{k-1}}{\Delta t} x(t_k), \quad t_d \in [t_{k-1}, t_k]$$

$$\frac{1}{D} \sum_{d=1:D} (\tilde{x}_d - \hat{x}_d)^2 + \alpha \sum_{...} \left(\frac{\Phi(x_{i+1}) - 2\Phi(x_i) + \Phi(x_{i-1})}{\Delta x^2} \right)^2 \Delta x \rightarrow \min_{\mathbf{x}_k, \mathbf{a}, \mathbf{b}, \boldsymbol{\omega}} .$$

Нелинейная задача математического программирования

Дискретизация: кусочно-линейная функция F + сетки

Внешнюю функцию аппроксимируем кусочно-линейной (**PWL**) с неизвестными значениями в изломах, “внутреннюю” — сеткой значений.

$$x(t) \approx \{(\mathbf{x}(\mathbf{t}_k), t_k) : k=0:N_t\},$$
$$F(x) \approx \{(\mathbf{F}_i, x_i) : i=0:N_x\}, \mathbf{F}_i = \mathbf{F}(\mathbf{x}_i).$$

$$x_i = x_{\mathbf{L}} + i \cdot \Delta x, \Delta x = \frac{x_{N_x} - x_{\mathbf{L}}}{N_x},$$
$$t_k = t_{\mathbf{L}} + k \cdot \Delta t, \Delta t = \frac{t_{N_t} - t_{\mathbf{L}}}{N_t}$$

Проблема: для PWL-функции нужны «специальные» способы ее записи в конечномерных задачах математического программирования...

Первый способ: «сглаживание» её представления в виде суммы негладких функций $[z]_+ = \max\{0, z\}$.

$$\hat{F}(x) = F_0 + (A_1 - A_0)(x - x_0) + \sum_{i=2}^{N_x} (A_i - A_{i-1}) [x - x_{i-1}]_+$$

$$A_i = \frac{F_i - F_{i-1}}{x_i - x_{i-1}}$$

$$[z]_+ \approx \frac{1}{2} \left(z + \frac{z^2}{\sqrt{z^2 + \varepsilon^2}} \right)$$

$$[z]_+ \approx \frac{1}{2} \left(z + \sqrt{z^2 + \varepsilon^2} \right)$$

(CHKS)

$$[z]_+ \approx z + \varepsilon \cdot \ln \left(\exp \left(\frac{-z}{\varepsilon} \right) + 1 \right)$$

(NWS)

CHKS — Chen-Harker-Kanzow-Smale,

NWS - Neural Networks Smoothing

Дискретизация: SPWL-функция на сетке (приведенная)

Представление на основе CHKS-сглаживания с параметром ε можно привести к следующему виду (в т.ч. и для неравномерной сетки !)

$$\begin{aligned}\tilde{F}(x) = & F_0 - A_1 x_0 + \frac{1}{2} (A_1 + A_{N_x}) \cdot x + \\ & + \frac{1}{2} \sum_{i=2}^{N_x} (A_i - A_{i-1}) \left(\sqrt{(x - x_{i-1})^2 + \varepsilon^2} - x_{i-1} \right) = \\ & \frac{1}{2} \left(F_0 + A_1 (x - x_0) + F_{N_x} + A_{N_x} (x - x_{N_x}) \right) + \\ & + \frac{1}{2} \sum_{i=2}^{N_x} (A_i - A_{i-1}) \sqrt{(x - x_{i-1})^2 + \varepsilon^2}\end{aligned}$$

$$A_i \doteq \frac{F_i - F_{i-1}}{x_i - x_{i-1}}$$

Для случая равномерной сетки

$$A_i \doteq \frac{F_i - F_{i-1}}{x_i - x_{i-1}} = \frac{F_i - F_{i-1}}{\Delta x}$$

$$\begin{aligned}\tilde{F}(x) = & \frac{1}{2} (F_0 + A_1 (x - x_0) + F_{N_x} + A_{N_x} (x - x_{N_x})) + \\ & + \frac{1}{2} \sum_{i=2}^{N_x} \left(\frac{F_i - 2F_{i-1} + F_{i-2}}{\Delta x} \right) \sqrt{(x - x_{i-1})^2 + \varepsilon^2}\end{aligned}$$

Везде алгебраические выражения

Дискретизация автономного ДУ: SPWL-функция на сетке (1)

$$\begin{aligned} \tilde{F}(x) = & \frac{1}{2} \left(F_0 + A_1(x - x_0) + F_{N_x} + A_{N_x}(x - x_{N_x}) \right) + \\ & + \frac{1}{2} \sum_{i=2}^{N_x} (A_i - A_{i-1}) \sqrt{(x - x_{i-1})^2 + \varepsilon^2} \\ \left\{ \begin{array}{l} \frac{1}{D} \sum_{d=1}^D (\tilde{x}_d - \hat{x}_d)^2 + \alpha \sum_{i=1}^{N_x-1} \left(\frac{F_{i+1} - 2F_i + F_{i-1}}{\Delta x^2} \right)^2 \Delta x \rightarrow \min_{x(t_k), \mathbf{F}_i = \mathbf{F}(x_i)}, \\ \frac{x(t_k) - x(t_{k-1})}{\Delta t} = \tilde{F} \left(\frac{x(t_k) + x(t_{k-1})}{2} \right), k = 1:N_t \quad \text{или} \\ \frac{x(t_{k+1}) - 2x(t_k) + x(t_{k-1})}{\Delta t^2} = \tilde{F}(x(t_k)), k = 1:(N_t - 1) \\ x_L \leq x(t_k) \leq x_U, F_L \leq F_i \leq F_U. \end{array} \right. \end{aligned}$$

интервальные ограничения на
переменные важны для алгоритмов
глобальной оптимизации

- + Здесь вполне эффективны алгоритмы поиска локального оптимума. В SvF применяется решатель Ipopt (Interior Point Opt.), <https://github.com/coin-or/Ipopt> (Eclipse Public License)
- Ограничения-равенства с “квадратичными” выражениями трудны для алгоритмов глобальной оптимизации. Мы применяем SCIP, www.scipopt.org (Apache 2.0 License с 4 Ноября 2022)

Дискретизация автономного ДУ: SPWL-функция на сетке (2)

$$\begin{aligned}\tilde{F}(x) = & \frac{1}{2} \left(F_0 + A_1(x - x_0) + F_{N_x} + A_{N_x}(x - x_{N_x}) \right) + \\ & + \frac{1}{2} \sum_{i=2}^{N_x} (A_i - A_{i-1}) \sqrt{(x - x_{i-1})^2 + \varepsilon^2}\end{aligned}$$

$A_i \doteq \frac{F_i - F_{i-1}}{x_i - x_{i-1}}$

Гладкая функция при $\varepsilon > 0$. Если ввести доп. переменные $\xi_i = \sqrt{(x - x_{i-1})^2 + \varepsilon^2}$ и ограничения равенства $\xi_i^2 = (x - x_{i-1})^2 + \varepsilon^2$, то получим **полиномы 2-й степени (по всем неизвестным)**. В критерии — будет также многочлен 2-й степени.

$$\left\{ \begin{array}{l} \frac{1}{D} \sum_{d=1}^D (\tilde{x}_d - \hat{x}_d)^2 + \alpha \sum_{i=1}^{N_x-1} \left(\frac{F_{i+1} - 2F_i + F_{i-1}}{\Delta x^2} \right)^2 \Delta x \rightarrow \min_{\boldsymbol{x}(t_k), \boldsymbol{F}_i = \boldsymbol{F}(\boldsymbol{x}_i)}, \\ \frac{x(t_k) - x(t_{k-1})}{\Delta t} = \tilde{F} \left(\frac{x(t_k) + x(t_{k-1})}{2} \right), k = 1:N_t \\ \frac{x(t_{k+1}) - 2x(t_k) + x(t_{k-1})}{\Delta t^2} = \tilde{F}(x(t_k)), k = 1:(N_t-1) \\ x_L \leq x(t_k) \leq x_U, F_L \leq F_i \leq F_U. \end{array} \right.$$

или

интервальные ограничения на переменные важны для алгоритмов глобальной оптимизации

При $\varepsilon > 0$ получаем невыпуклую задачу математического программирования с гладкими функциями в критерии и ограничениях.

SOS2 для «точного» представления PWL-функций

Кусочно-линейную функцию $F(x): \mathbb{R}^1 \rightarrow \mathbb{R}^1$ (и не только!) можно представить на основе т. н. **специально упорядоченных наборов множеств (чисел)**.

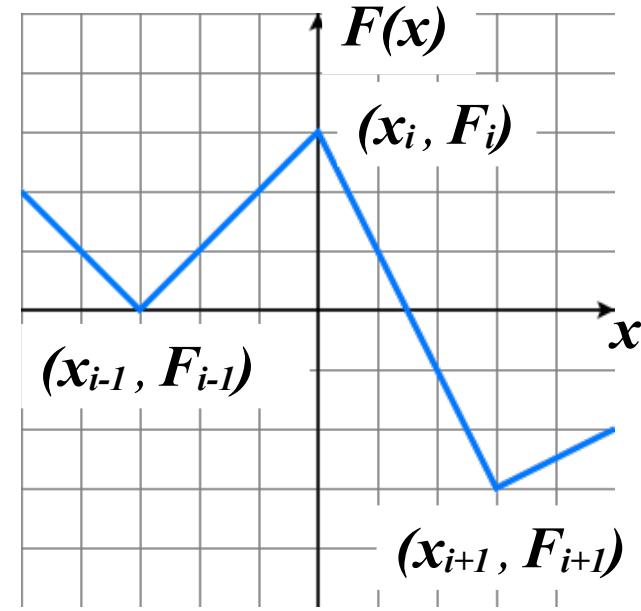
E. Beale and J. Tomlin "Special Facilities in General Mathematical Programming System for Non-Convex Problems Using Ordered Sets of Variables" (1970)

Для функции $F(x): \mathbb{R}^1 \rightarrow \mathbb{R}^1$ требуется т. н. SOS2 (Special Ordered Sets type 2)

$$\lambda \in \mathbb{R}^{N_x+1}, \lambda \in \mathbf{SOS2}(N_x) \Leftrightarrow \{ \lambda_i \neq 0 \text{ и } \lambda_j \neq 0 \Rightarrow |i-j|=1 \ (\forall i, j \in 0:N_x) \}$$

Только два соседних коэффициента могут быть оба отличны от нуля

$$\left\{ \begin{array}{l} \lambda \in \mathbf{SOS2}(N_x), \\ \lambda \in \Delta(N_x) \doteq \left\{ \sum_{j=0}^{N_x} \lambda_j = 1, \lambda_i \geq 0 \right\}, \\ \hat{F}(x) = \sum_{j=0}^{N_x} \lambda_j F_i, \\ x = \sum_{j=0}^{N_x} \lambda_j x_i \end{array} \right.$$



SOS2 для представления PWL-функций. Автономное ОДУ 1

Обратная задача в этом случае выглядит так: те же сетки по переменным t и x , но PWL-функция задана “точно” на основе SOS2-ограничений.

Для каждой точки t_k — нужен свой набор SOS2-переменных λ (!).

Для ОДУ 1-го порядка

$$\frac{1}{D} \sum_{d=1}^D (\tilde{x}_d - \hat{x}_d)^2 + \alpha \sum_{i=1}^{N_x-1} \left(\frac{F_{i+1} - 2F_i + F_{i-1}}{\Delta x^2} \right)^2 \Delta x \rightarrow \min_{x(t_k), F_i = F(x_i)},$$

$$\frac{x(t_k) - x(t_{k-1})}{\Delta t} = \sum_{i=0}^{N_x} \lambda_i^{(k)} F_i \quad (k=1:N_t),$$

$$\frac{x(t_k) + x(t_{k-1})}{2} = \sum_{i=0}^{N_x} \lambda_i^{(k)} x_i \quad (k=1:N_t),$$

$$\sum_{i=0}^{N_x} \lambda_i^{(k)} = 1, \lambda_i^{(k)} \geq 0 \quad (i=0:N_x)$$

$$\lambda^{(k)} \in \text{SOS2}(N_x),$$

$$x_L \leq x(t_k) \leq x_U, F_L \leq F_i \leq F_U.$$

$$\boxed{\frac{x(t_k) - x(t_{k-1})}{\Delta t} = \hat{F} \left(\frac{x(t_k) + x(t_{k-1})}{2} \right)}$$

SOS2 в теории дизъюнктных ограничений и в практике

Понятие SOS2 имеет важные обобщения в прикладной оптимизации, в задачах с т. н. «**disjunctive constraints**» («выполнено либо это, либо ... »). Например, статья Vielma J. P. , Nemhauser G. L. Modeling disjunctive constraints with a logarithmic number of binary variables and constraints (2011) цитируется >330 раз. Намечены возможности применения для PWL-функций 2x и более переменных.

SOS2 — аналог объединения
ребер симплекса в \mathbb{R}^{N_x+1}

$$\Delta^{N_x+1} \doteq \left\{ \lambda \in \mathbb{R}^{N_x+1} : \lambda \geq 0, \sum_{j=0:N_x} \lambda_j = 1 \right\}$$

$$S[i-1, i] \doteq \left\{ \lambda \in \Delta^{N_x+1} : \lambda_{i-1} + \lambda_i = 1 \right\} (i=1:N_x)$$

$$SOS2(N_x) \doteq \bigcup_{i=0:N_x} S[i-1, i] \Rightarrow \left\{ \lambda_i \neq 0 \text{ и } \lambda_j \neq 0 \Rightarrow |i-j| \leq 1 (\forall i, j \in 0:N_x) \right\}$$

Основные коммерческие и «свободные» солверы глобальной оптимизации поддерживают работу с SOS2-переменными в алгоритмах Ветвей-и-Границ (BnB): Fico Xpress, Gurobi, Cplex, COPT, SCIP.

Проблема PWL-представление функции 2, 3 переменных

Форма записи кусочно-линейных функций для задач математического программирования популярная тема вычислительной математики

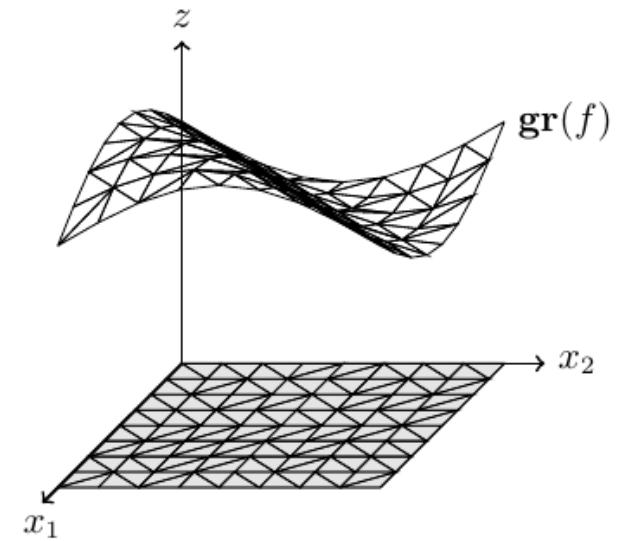
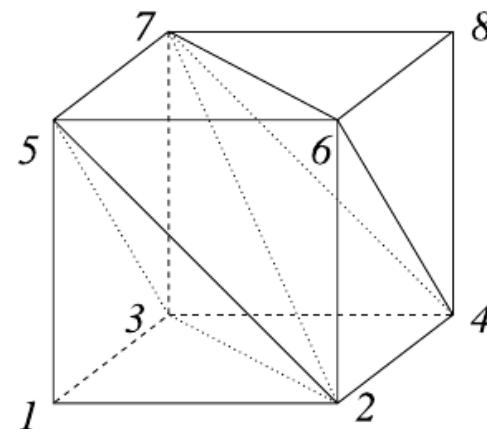
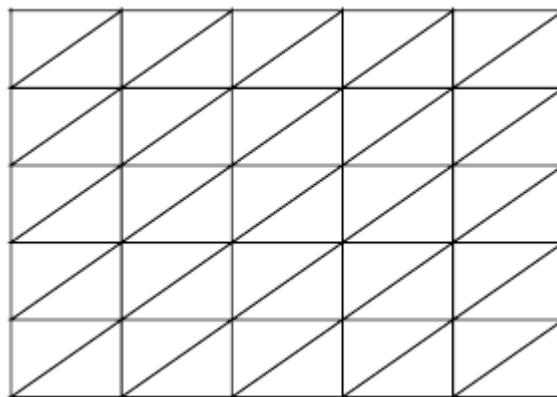
Misener, R. and Floudas, C.A., 2010. Piecewise-linear approximations of multidimensional functions. Journal of optimization theory and applications

Vielma, J.P. and Nemhauser, G.L., 2011. Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. Mathematical Programming

Huchette, J. and Vielma, J.P., 2023. Nonconvex piecewise linear functions: Advanced formulations and simple modeling tools. Operations Research

+ Точные выражения на основе 2D, 3D «триангуляций»

-- Сложность и «нестандартность» (для применяемых нами решателей)



PWL/SPWL-функции двух переменных

Был предложен упрощенный, “неточный” подход на основе «рекуррентного» применения PWL/SPWL аппроксимации функций одного переменного

$$\{(x_i, y_j) : i=0:N_x, j=0:N_y\};$$

$$F(x, y) \approx \{(\hat{F}_{ij}, x_i, y_j) : i=0:N_x, j=0:N_y\}.$$

Значение $F(x, y)$ вычисляется так:

1. для всех y_j строятся PWL- или SPWL-функции $\hat{F}_j(x)|\tilde{F}_j(x)$;

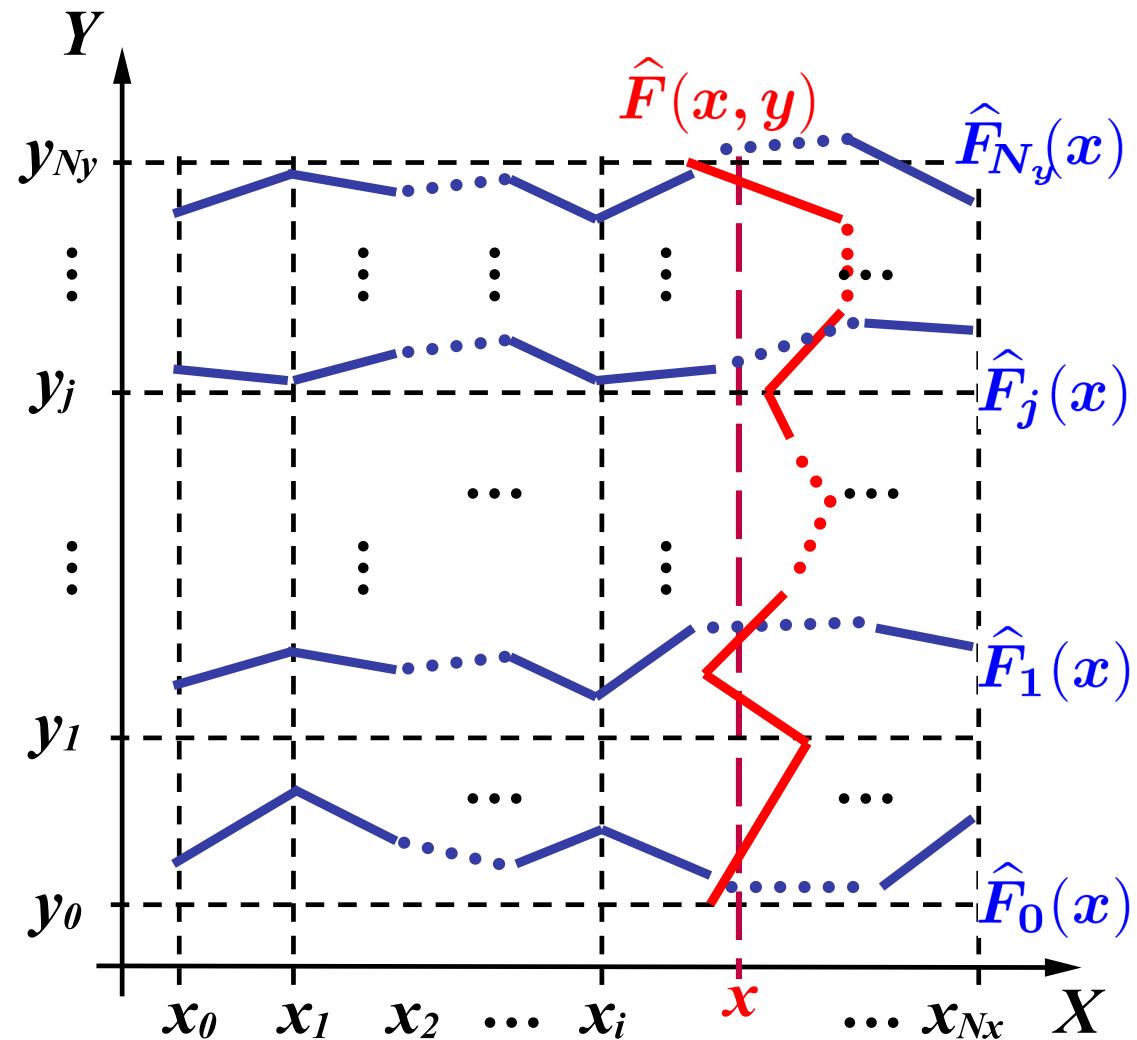
2. для заданного x строится PWL|SPWL-функция по y при фиксированном x

$$\hat{F}(x, y)|\tilde{F}(x, y)$$

по сетке значений

$$\{\hat{F}_j(x)|\tilde{F}_j(x) : j=0:N_y\}.$$

Для SvF-метода не требуются точные значения, возможны приближенные! Регуляризация !!!



SmoothedPWL-представление функции $F(x,y):R^2 \rightarrow R^1$

$$F(\mathbf{x}, \mathbf{y}) \approx \{(F_{ij}, x_i, y_j) : i=0:N_x, j=0:N_y\}, \tilde{F}(x, y) - ?$$

$$\begin{aligned} \tilde{F}_j(x) = & \frac{1}{2} \left(F_{0j} + A_{1j}(x-x_0) + F_{N_x,j} + A_{N_x,j}(x-x_{N_x}) \right) + \\ & + \frac{1}{2} \sum_{i=2}^{N_x} (A_{ij} - A_{i-1,j}) \sqrt{(x-x_{i-1})^2 + \varepsilon^2} \quad (j=0:N_y). \end{aligned}$$

Везде алгебраические выражения

$$A_{ij} \doteq \frac{F_{ij} - F_{i-1,j}}{x_i - x_{i-1}}$$

$$\tilde{A}_j(x) \doteq \frac{\tilde{F}_j(x) - \tilde{F}_{j-1}(x)}{y_j - y_{j-1}}$$

$$\begin{aligned} \tilde{F}(\mathbf{x}, \mathbf{y}) = & \frac{1}{2} \sum_{j=2}^{N_y} \left(\tilde{A}_j(x) - \tilde{A}_{j-1}(x) \right) \sqrt{(y-y_{j-1})^2 + \varepsilon^2} + \\ & + \frac{1}{2} \left(\tilde{F}_0(x) + \tilde{A}_1(x)(y-y_0) + \tilde{F}_{N_y}(x) + \tilde{A}_{N_y}(x)(y-y_{N_y}) \right) \end{aligned}$$

Оценим “степень нелинейности” полученного представления. Для функции одного переменного SPWL была билинейной. $\xi_i = \sqrt{(x-x_i)^2 + \varepsilon^2}$, $\eta_j = \sqrt{(y-y_j)^2 + \varepsilon^2}$

$$\tilde{F}_j(x) \sim (F_{ij}, x, \xi_i, F_{ij} \cdot x, F_{ij} \cdot \xi_i) \quad \tilde{A}_j(x) \sim (\tilde{F}_j(x), \tilde{F}_{j-1}(x))$$

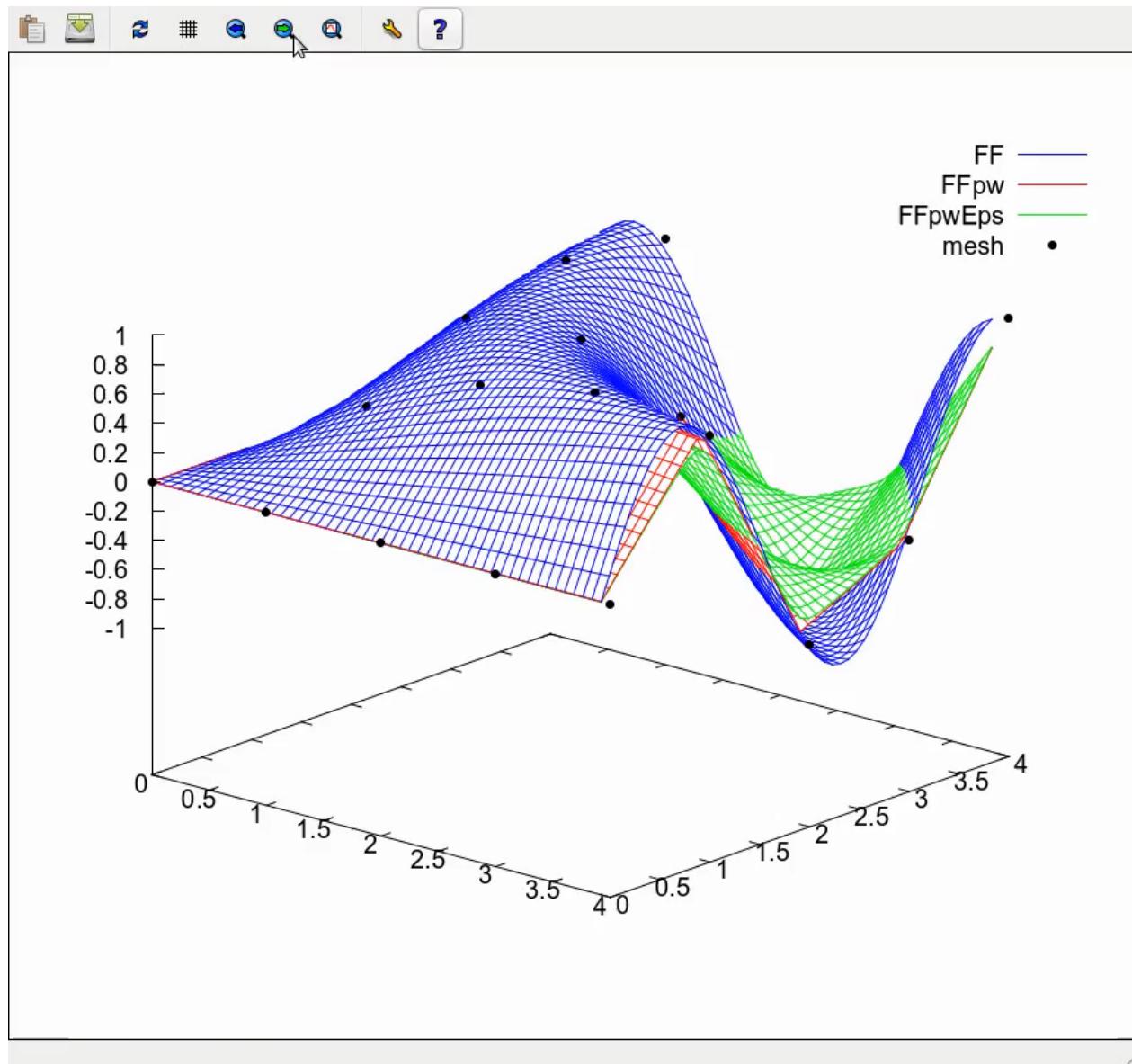
$$\tilde{F}(x, y) \sim (F_{ij}, x, \xi_i, F_{ij} \cdot x, F_{ij} \cdot \xi_i) \cdot (y + \eta_j)$$

Эти оценки показывают, что для функции двух переменных сглаженная кусочно-линейная аппроксимация приводит к многочленам 3й степени.

Пример SPWL-представления функции $F(x,y)$

Демонстрация SPWL-аппроксимации “произвольной” $F(x,y)$.

$$F(x, y) = \sin(0.5 \cdot x \cdot y), Nx=Ny=4, \varepsilon=0.1$$



SPWL-дискретизация ОДУ $\frac{dx}{dt} | \frac{d^2x}{dt^2} = F(x(t), y(t), t)$

$$(x(t), y(t)) \sim \{(x(t_k), y(t_k), t_k) : k=0:N_t\};$$

$$F(x, y, t) \sim \left\{ \left(\mathbf{F}_{ij}^{(k)}, x_i, y_j \right) : i=0:N_x, j=0:N_y, k=0:N_t \right\}, \mathbf{F}_{ij}^{(k)} = F(x_i, y_j, t_k).$$

$$A_{ij}^{(k)} \doteq \frac{F_{ij}^{(k)} - F_{i-1,j}^{(k)}}{x_i - x_{i-1}}, \tilde{F}_j^{(k)}(x) = *, \tilde{A}_j^{(k)}(x) \doteq \frac{\tilde{F}_j^{(k)}(x) - \tilde{F}_{j-1}^{(k)}(x)}{y_j - y_{j-1}}, \tilde{F}^{(k)}(x, y) = *$$

$$\text{Err}(\{\tilde{x}_d, x(t_k), y(t_k)\}) + R \left(\alpha_t, \alpha_x, \alpha_y, F_{ij}^{(k)}, N_t, N_x, N_y \right) \rightarrow \min_{\mathbf{x}(t_k), \mathbf{y}(t_k), \mathbf{F}_{ij}^{(k)}},$$

$$\frac{x(t_k) - x(t_{k-1})}{\Delta t} = \tilde{F}^{(k)} \left(\frac{x(t_k) + x(t_{k-1})}{2}, \frac{y(t_k) + y(t_{k-1})}{2} \right) \quad (k=1:N_t)$$

или

$$\frac{x(t_{k+1}) - 2x(t_k) + x(t_{k-1})}{\Delta t^2} = \tilde{F}^{(k)}(x(t_k), y(t_k), k=1:(N_t-1))$$

$$x_L \leq x(t_k) \leq x_U, y_L \leq y(t_k) \leq y_U, F_L \leq F_{ij}^{(k)} \leq F_U.$$

Невыпуклая задача математического программирования с многочленами **не выше 3й степени** в критерии и ограничениях.

SOS2 представление PWL-функции $F(x,y):R^2 \rightarrow R^1$

$F(x, y) \sim \{(\mathbf{F}_{ij}, x_i, y_j) : i=0:N_x, j=0:N_y\}$, $\hat{F}(x, y)$ - ?

$$\{\boldsymbol{\lambda}_i : i=0:N_x\}, \{\boldsymbol{\mu}_j : j=0:N_y\},$$

$$\hat{F}(x, y) = \sum_{j=0}^{N_y} \boldsymbol{\mu}_j \sum_{i=0}^{N_x} \boldsymbol{\lambda}_i \mathbf{F}_{ij}, \quad \hat{F}_j(x) = \sum_{i=0}^{N_x} \boldsymbol{\lambda}_i(x) \mathbf{F}_{ij}$$

$$(\boldsymbol{\mu}_j)_{j=0}^{N_y} \geq 0, \sum_{j=0}^{N_y} \boldsymbol{\mu}_j = 1, (\boldsymbol{\mu}_j)_{j=0}^{N_y} \in \text{SOS2}(N_y),$$

$$(\boldsymbol{\lambda}_i)_{i=0}^{N_x} \geq 0, \sum_{i=0}^{N_x} \boldsymbol{\lambda}_i = 1, (\boldsymbol{\lambda}_i)_{i=0}^{N_x} \in \text{SOS2}(N_x),$$

$$y = \sum_{j=0}^{N_y} \boldsymbol{\mu}_j y_j, \quad x = \sum_{i=0}^{N_x} \boldsymbol{\lambda}_i x_i \quad (j=0:N_y)$$

Итого, на одну точку (x, y) нужно:

$N_x + N_y + 2$ вспомогательных переменных ($\boldsymbol{\lambda}_i, \boldsymbol{\mu}_j$),

по одному ограничению $\text{SOS2}(N_x)$ и $\text{SOS2}(N_y)$.

SOS2-дискретизация $\frac{dx}{dt} = F(x(t), y(t))$

$$(x(t), y(t)) \sim \{(\mathbf{x}(t_k), \mathbf{y}(t_k), t_k) : k=0:N_t\};$$

$$F(x, y) \sim \{(\mathbf{F}_{ij}, x_i, y_j) : i=0:N_x, j=0:N_y\}, \mathbf{F}_{ij} = F(x_i, y_j).$$

$$\left\{ \boldsymbol{\lambda}_i^{(k)} : i=0:N_x \right\}, \left\{ \boldsymbol{\mu}_j^{(k)} : i=0:N_y \right\}, k=0:N_t$$

$$\frac{\mathbf{x}(t_{k+1}) - \mathbf{x}(t_k)}{\Delta t} = \sum_{j=0}^{N_y} \boldsymbol{\mu}_j^{(k)} \sum_{i=0}^{N_x} \boldsymbol{\lambda}_i^{(k)} \mathbf{F}_{ij},$$

$$\left(\mu_j^{(k)} \right)_{j=0}^{N_y} \geq 0, \sum_{j=0}^{N_y} \mu_j^{(k)} = 1, \left(\mu_j^{(k)} \right)_{j=0}^{N_y} \in \text{SOS2}(N_y),$$

$$\forall j=\overline{0, N_y} : \lambda_{ij}^{(k)} \geq 0 \quad (i=0:N_x), \sum_{i=0}^{N_x} \lambda_{ij}^{(k)} = 1, \left(\lambda_{ij}^{(k)} \right)_{i=0}^{N_x} \in \text{SOS2}(N_x),$$

$$\frac{\mathbf{y}(t_{k+1}) + \mathbf{y}(t_k)}{2} = \sum_{j=0}^{N_y} \mu_j^{(k)} y_j, \quad \frac{\mathbf{x}(t_{k+1}) + \mathbf{x}(t_k)}{2} = \sum_{i=0}^{N_x} \lambda_{ij}^{(k)} x_i \quad (j=0:N_y)$$

Для каждой точки t_k нужны свои наборы **Nx+Ny+2** вспомогательных переменных (λ_i, μ_j) и по одному ограничению $\text{SOS2}(N_x)$ и $\text{SOS2}(N_y)$.

Нелинейность 3-й степени $\boldsymbol{\mu}_j^{(k)} \cdot \boldsymbol{\lambda}_i^{(k)} \cdot \mathbf{F}_{ij}$

Если есть еще уравнение $dy/dt = G(x, y)$, то для него — те же представления.

Какие способы дискретизации применялись

$$F(x) \approx \mathcal{P}(\mathbf{c}, x) = \sum_{p=0}^P \mathbf{c}_p \cdot x^p$$

Полином.
Фурье

$$F(x) \approx \Phi(\mathbf{a}, \mathbf{b}, \boldsymbol{\omega}, x) =$$

$$= \sum_{p=0}^P \mathbf{a}_p \cdot \cos(p\boldsymbol{\omega}x) + \sum_{p=1}^P \mathbf{b}_p \cdot \sin(p\boldsymbol{\omega}x)$$

Сеточное представление $F(x)$

$$\begin{aligned} \tilde{F}(x) &= \frac{1}{2} \sum_{i=2}^{N_x} (\mathbf{A}_i - \mathbf{A}_{i-1}) \sqrt{(x - x_{i-1})^2 + \varepsilon^2} + \\ &+ \frac{1}{2} \left(\mathbf{F}_0 + \mathbf{A}_1 \cdot (x - x_0) + \mathbf{F}_{N_x} + \mathbf{A}_{N_x} \cdot (x - x_{N_x}) \right) \end{aligned}$$

$$\begin{aligned} x(t) &\approx \{(\mathbf{x}(t_k), t_k) : k=0:N_t\}, \\ t_k &= t_{\mathbf{L}} + k \cdot \Delta t, \Delta t = \frac{t_{N_t} - t_{\mathbf{L}}}{N_t}, \\ x_i &= x_{\mathbf{L}} + i \cdot \Delta x, \Delta x = \frac{x_{N_x} - x_{\mathbf{L}}}{N_x}. \end{aligned}$$

$$\begin{aligned} F(x) &\approx \{(\mathbf{F}_i, x_i) : i=0:N_x\} \\ \mathbf{A}_i &\doteq \frac{\mathbf{F}_i - \mathbf{F}_{i-1}}{x_i - x_{i-1}} \end{aligned}$$

SPWL
Smoothed Piecewise Linear

$$\begin{cases} \lambda \in \text{SOS2}(N_x), \sum_{j=0}^{N_x} \lambda_j = 1, \lambda_j \geq 0, \\ \hat{F}(x) = \sum_{j=0}^{N_x} \lambda_j \mathbf{F}_j, \quad x = \sum_{j=0}^{N_x} \lambda_j x_i \end{cases}$$

SOS2

Для каждого x - отдельный SOS2-вектор λ

Число переменных у Р/Ф-аппроксимации $F(x,y)$ (1)

$$F(x,y) \approx \mathcal{P}(\mathbf{c}, x, y) = \sum_{p+q \leqslant P}^{P,P} \mathbf{c}_{pq} x^p y^q$$

$$|\mathbf{c}| = \frac{(P+2)(P+1)}{2}$$

нелинейность $\sim P+1$

Вариант частичной суммы кратного ряда Фурье

$$\begin{aligned} F(x,y) &\approx \Phi(\mathbf{a}, \mathbf{b}, \mathbf{c}, \boldsymbol{\omega}_x, \boldsymbol{\omega}_y, x, y) = \sum_{p+q \leqslant P}^{P,P} \mathbf{c}_{pq} \cos(p\boldsymbol{\omega}_x x) \sin(q\boldsymbol{\omega}_y y) + \\ &+ \sum_{p+q \leqslant P}^{P,P} \left\{ \mathbf{a}_{pq} \cos(p\boldsymbol{\omega}_x x) \cos(q\boldsymbol{\omega}_y y) + \mathbf{b}_{pq} \sin(p\boldsymbol{\omega}_x x) \sin(q\boldsymbol{\omega}_y y) \right\} \end{aligned}$$

$$|\mathbf{c}| + |\mathbf{a}| + |\mathbf{b}| + |\boldsymbol{\omega}| = 3 \frac{(P+2)(P+1)}{2} + 2$$

нелинейность на $[0,1] \times [0,1] \sim P^2$

$$\max_{[0,1]} \frac{d^2}{dx^2} e^{i \cdot P \cdot x}$$

Число переменных в SPWL аппроксимации $F(x,y)$ (2)

Сеточное представление повторное применение сглаживания кусочно-линейных функций

$$\{(x_i, y_j) : i=0:N_x, j=0:N_y\}; F(x, y) \approx \{(\mathbf{F}_{ij}, x_i, y_j) : i=0:N_x, j=0:N_y\}.$$

$$\begin{aligned}\tilde{F}_j(x) = & \frac{1}{2} \left(F_{0j} + A_{1j}(x - x_0) + F_{N_x,j} + A_{N_x,j}(x - x_{N_x}) \right) + \\ & + \frac{1}{2} \sum_{i=2}^{N_x} (A_{ij} - A_{i-1,j}) \sqrt{(x - x_{i-1})^2 + \varepsilon^2} \quad (j=0:N_y).\end{aligned}$$

Везде алгебраические выражения

$$A_{ij} \doteq \frac{\mathbf{F}_{ij} - \mathbf{F}_{i-1,j}}{x_i - x_{i-1}}$$

$$\tilde{A}_j(x) \doteq \frac{\tilde{F}_j(x) - \tilde{F}_{j-1}(x)}{y_j - y_{j-1}}$$

$$\begin{aligned}\tilde{F}(\mathbf{x}, \mathbf{y}) = & \frac{1}{2} \sum_{j=2}^{N_y} \left(\tilde{A}_j(x) - \tilde{A}_{j-1}(x) \right) \sqrt{(y - y_{j-1})^2 + \varepsilon^2} + \\ & + \frac{1}{2} \left(\tilde{F}_0(x) + \tilde{A}_1(x)(y - y_0) + \tilde{F}_{N_y}(x) + \tilde{A}_{N_y}(x)(y - y_{N_y}) \right)\end{aligned}$$

Для функции двух переменных сглаженная кусочно-линейная аппроксимация приводит к **многочленам 3й степени в ограничениях-равенствах**.

Число переменных $N_x * N_y$ (число точек сетки для $F(x,y)$). Но для (неавтономных) ОДУ, для каждого момента времени нужен свой набор \mathbf{F}_{ij} — в итоге **для 20 точек по времени и сетки 10x10 будет 2000 переменных**

Сочетание 1-дискретизаций для $F(x,y)$

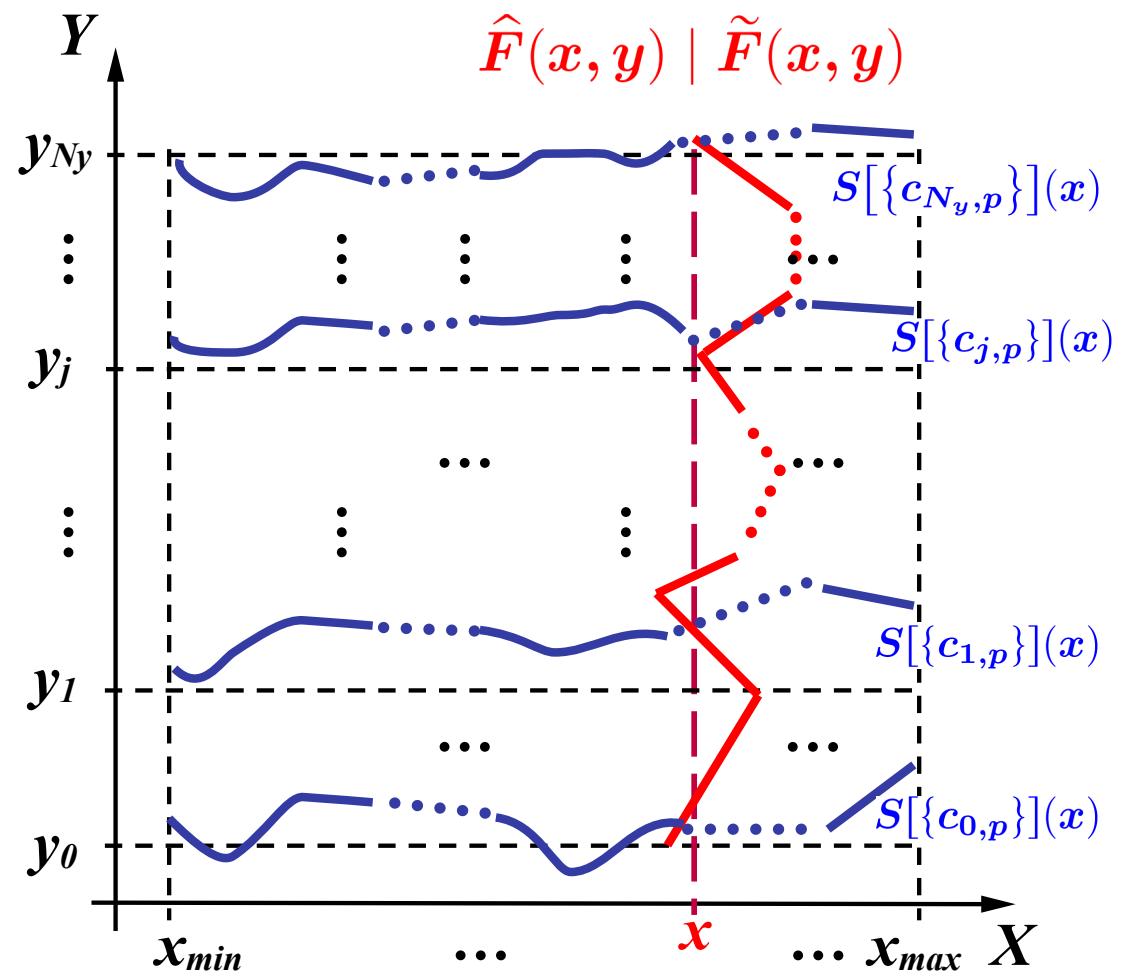
Новая, «неточная», 2-дискретизация: на основе «сочетания» различных 1-дискретизаций и PWL/SPWL аппроксимации функций одного переменного

$$\left\{ \left(S(\{\mathbf{c}_{j,p}\}, x), y_j \right) : j=0:N_y \right\}, S(\{\mathbf{c}_{j,p}\}, x) = \mathcal{P}(\mathbf{c}_{j,p}, x) | \Phi(a_j, b_j, \omega_j, x)$$

Значение $F(x,y)$ вычисляется так:

1. Для всех y_j имеем суммы, $S(\{\mathbf{c}_{j,p}\}, x)$ как функции x
2. Для заданного \mathbf{x} строится PWL|SPWL-функция по y при фиксированном \mathbf{x}
 $\hat{F}(\mathbf{x}, y) | \tilde{F}(\mathbf{x}, y)$
 по сетке значений $\{S(\{\mathbf{c}_{j,p}\}, \mathbf{x})\} : j=0:N_y\}$

Для SvF не требуются точные значения $F(x,y)$, возможны приближенные! Регуляризация !!!



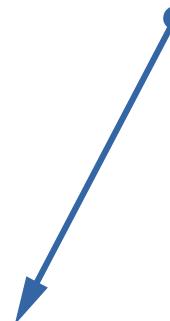
Число переменных в «новой» схеме дискретизации $F(x,y)$

Суммы с неизвестными коэффициентами и сглаживание кусочно-линейных 1-функций

$$\{y_j:j=0:N_y\}; F(x,y) \approx \{(S(\{\mathbf{c}_{j,p}\},x), y_j) : j=0:N_y\}.$$

$$\{(S(\{\mathbf{c}_{j,p}\},x), y_j) : j=0:N_y\} \rightarrow A_j(\mathbf{c}_j, x) \doteq \frac{S(\{\mathbf{c}_{j,p}\},x) - S(\{\mathbf{c}_{j-1,p}\},x)}{y_j - y_{j-1}}$$

Везде алгебраические выражения (SPWL!)



$$\begin{aligned} \tilde{F}(\{\mathbf{c}_{j,p}\}, x, y) &= \frac{1}{2} \sum_{j=2}^{N_y} (A_j(\mathbf{c}_j, x) - A_{j-1}(\mathbf{c}_{j-1}, x)) \sqrt{(y - y_{j-1})^2 + \varepsilon^2} + \\ &+ \frac{1}{2} \left(S(\{\mathbf{c}_0\}, x) + A_1(\mathbf{c}_1, x)(y - y_0) + S(\{\mathbf{c}_{N_y}\}, x) + A_{N_y}(\mathbf{c}_{N_y}, x)(y - y_{N_y}) \right) \end{aligned}$$

«Нелинейность» ограничений оценивается как $P+1$...

Число переменных P^*N_y (коэффициентов $\{\mathbf{c}_{j,p}\}$). Было N_x*N_y , но теперь нет сетки по x (с одной оговоркой, см. далее). Для (неавтономных) ОДУ, для каждого момента времени нужен свой набор коэффициентов $\{\mathbf{c}_{j,p}^t\}$.

Еще раз о важной роли регуляризации

В ходе расчетов (идентификации) функции $F(x,y)$ будут определяться коэффициентами $\{\mathbf{c}_{j,p}\}$.

$$\{y_j : j=0:N_y\}; F(x, y) \approx \{(S(\{\mathbf{c}_{j,p}\}, x), y_j) : j=0:N_y\}.$$

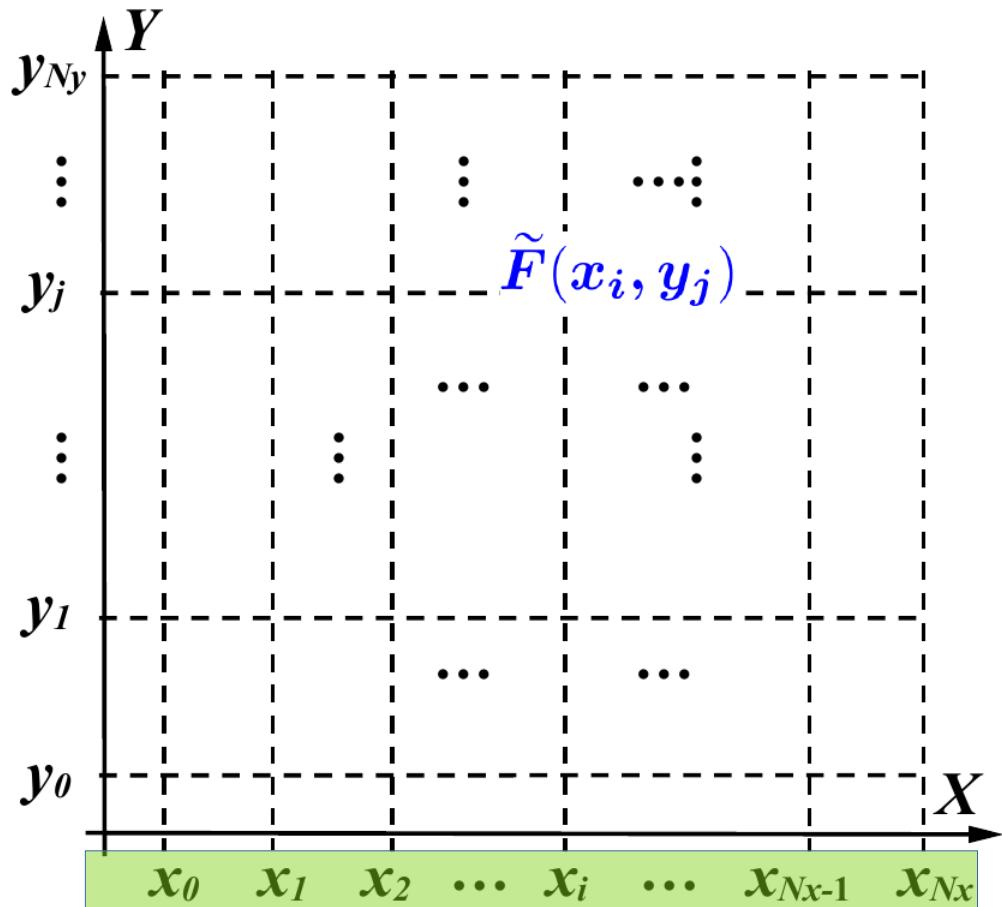
Формально, для соседних значений y_j значения коэффициентов никак не связаны. Чтобы они не «расходились» и порождали «гладкую» функцию $F(x,y)$ в критерий минимизации входит регуляризирующее слагаемое (нужна сетка по x !):

$$R(\alpha_x, \alpha_y, F(\cdot, \cdot)) =$$

$$\int_{y_L}^{y_U} \int_{x_L}^{x_U} \alpha_x^2 (F''_{xx}(x, y))^2 dx dy +$$

$$+ \int_{y_L}^{y_U} \int_{x_L}^{x_U} 2\alpha_x \alpha_y (F''_{xy}(x, y))^2 dx dy +$$

$$+ \int_{y_L}^{y_U} \int_{x_L}^{x_U} \alpha_y^2 (F''_{yy}(x, y))^2 dx dy$$



Разностные схемы для регуляризации

$$R(\alpha_x, \alpha_y, F(\cdot, \cdot)) \simeq \int_{y_L}^{y_U} \int_{x_L}^{x_U} \left(\alpha_x^2 \left(\tilde{F}_{xx}''(x, y) \right)^2 + 2\alpha_x \alpha_y \left(\tilde{F}_{xy}''(x, y) \right)^2 + \alpha_y^2 \left(\tilde{F}_{yy}''(x, y) \right)^2 \right) dx dy$$

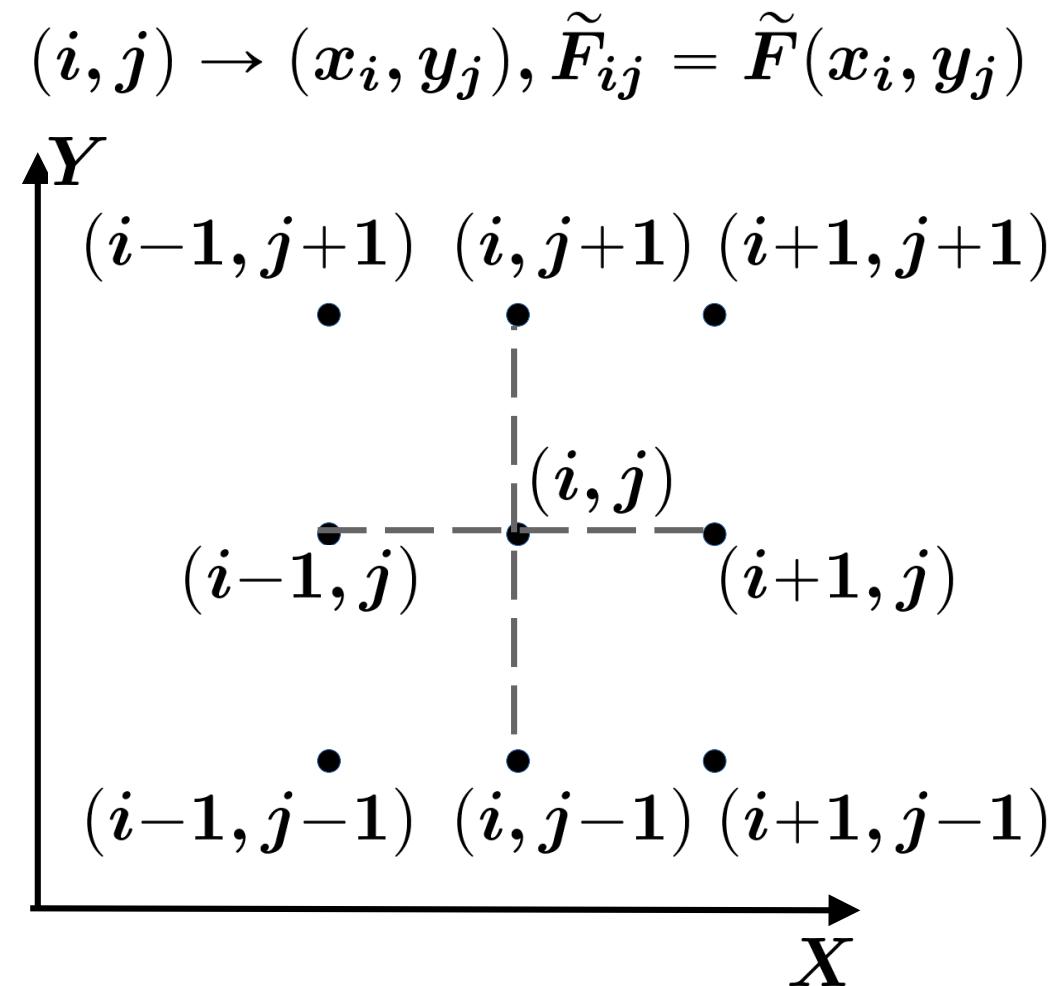
Для вычисления регуляризирующего слагаемого используем выражения повторных производных на основе “крестовых” разностных схем. Здесь — «прямой» крест.

$$F_{i+1,j} \simeq F_{i,j} + F'_{x,i,j} h_x + \frac{1}{2} F''_{xx,i,j} h_x^2,$$

$$F_{i-1,j} \simeq F_{i,j} - F'_{x,i,j} h_x + \frac{1}{2} F''_{xx,i,j} h_x^2$$

$$\tilde{F}_{xx,i,j}'' \simeq \frac{\tilde{F}_{i+1,j} - 2\tilde{F}_{i,j} + \tilde{F}_{i-1,j}}{h_x^2}$$

$$\tilde{F}_{yy,i,j}'' \simeq \frac{\tilde{F}_{i,j+1} - 2\tilde{F}_{i,j} + \tilde{F}_{i,j-1}}{h_y^2}$$

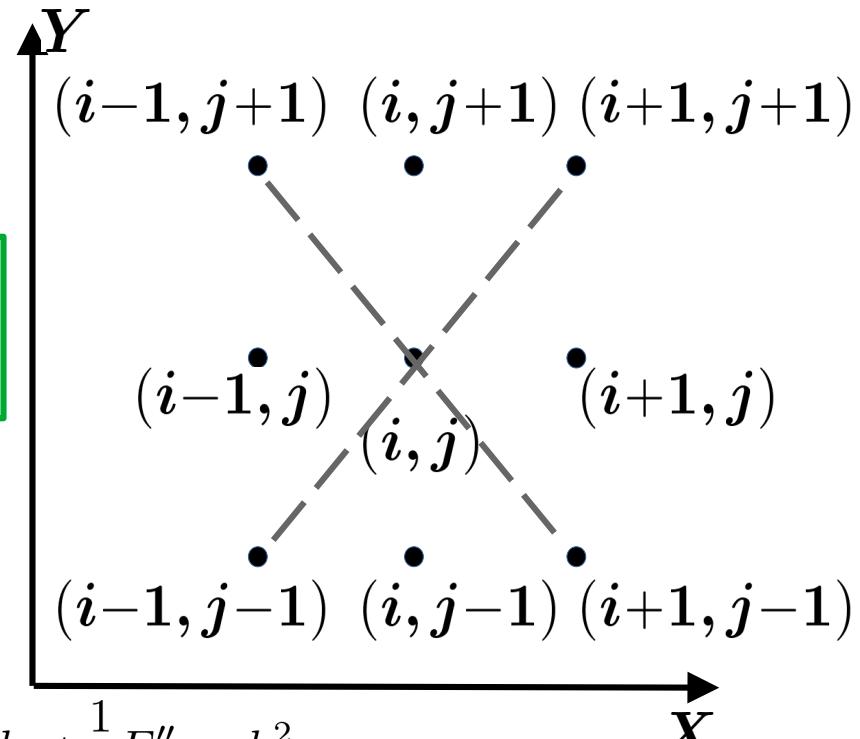


Разностные схемы для повторных производных (2)

Для регуляризующего слагаемого можно взять интегралы повторных производных на основе “крестовых” разностных схем. Здесь — «диагональный» крест

$$(i, j) \rightarrow (x_i, y_j), \tilde{F}_{ij} = \tilde{F}(x_i, y_j)$$

$$\tilde{F}_{xy,i,j}'' \simeq \frac{\tilde{F}_{i+1,j+1} - \tilde{F}_{i+1,j-1} - \tilde{F}_{i-1,j+1} + \tilde{F}_{i-1,j-1}}{4h_x h_y}$$



$$\begin{aligned}
 \oplus F_{i+1,j+1} &\simeq F_{i,j} + F'_{x,i,j} h_x + F'_{y,i,j} h_y + \frac{1}{2} F''_{xx,i,j} h_x^2 + F''_{xy,i,j} h_x h_y + \frac{1}{2} F''_{yy,i,j} h_y^2, \\
 \ominus F_{i+1,j-1} &\simeq F_{i,j} + F'_{x,i,j} h_x - F'_{y,i,j} h_y + \frac{1}{2} F''_{xx,i,j} h_x^2 - F''_{xy,i,j} h_x h_y + \frac{1}{2} F''_{yy,i,j} h_y^2, \\
 \ominus F_{i-1,j+1} &\simeq F_{i,j} - F'_{x,i,j} h_x + F'_{y,i,j} h_y + \frac{1}{2} F''_{xx,i,j} h_x^2 - F''_{xy,i,j} h_x h_y + \frac{1}{2} F''_{yy,i,j} h_y^2, \\
 \oplus F_{i-1,j-1} &\simeq F_{i,j} - F'_{x,i,j} h_x - F'_{y,i,j} h_y + \frac{1}{2} F''_{xx,i,j} h_x^2 + F''_{xy,i,j} h_x h_y + \frac{1}{2} F''_{yy,i,j} h_y^2. \\
 \ast \ast \ast \ast \simeq 0 & \quad 0 \quad 0 \quad 0 \quad 4F''_{xy,i,j} h_x h_y \quad 0
 \end{aligned}$$

$S(c,x) + \text{SOS2}$ представление функции $F(x,y):R^2 \rightarrow R^1$

$$\{y_j : j=0:N_y\}; F(x, y) \approx \{(S(\{\mathbf{c}_{j,p}\}, x), y_j) : j=0:N_y\} \quad \hat{F}(x, y) - ?$$

$$\{\boldsymbol{\mu}_j : i=0:N_y\},$$

$$\left\{ \begin{array}{l} \hat{F}(x, y) = \sum_{j=0}^{N_y} \boldsymbol{\mu}_j S(\{\mathbf{c}_{j,p}\}, x), \\ y = \sum_{j=0}^{N_y} \mu_j y_j, \\ (\mu_j)_{j=0}^{N_y} \geq 0, \sum_{j=0}^{N_y} \mu_j = 1, (\mu_j)_{j=0}^{N_y} \in \text{SOS2}(N_y). \end{array} \right.$$

Итого, на одну точку (x, y) :

сочетание SOS2 (по x) и SOS2 (по y) приводило к $Nx + Ny + 2$

вспомогательным переменным (λ_i, μ_j) и двум ограничениям $\text{SOS2}(Nx)$, $\text{SOS2}(Ny)$.

Сочетание SOS2 (по y) и $S(c, x)$ (по x) оставляет $Ny + 1$ SOS2-переменных μ_j

С другой стороны - «нелинейность» стала выше: $\sim P+1$, а было 3.

Применяемое ПО и проблемы производительности

SvF-технология, реализована на языке Python.

Подготовка данных и взаимодействие с солверами на основе пакета **Puomo**,
Python Optimization Modelling Objects, www.puomo.org

До недавнего времени основной солвер **Iloopt**, <https://github.com/coin-or/Iloopt> (Eclipse Public License). Реализует метод «внутренней точки» для поиска локальных решений в задачах мат. программирования с гладкими функциями.

Для удалённого доступа к солверам сервис на основе платформе **Everest**,
<http://everest.distcomp.org/>, <https://optmod.distcomp.org/apps/vladimirv/SSOP>

Для применение алгоритма глобальной оптимизации (BnB) сейчас используется солвер **SCIP**, www.scipopt.org (Apache 2.0 License).

Основная проблема: для задач «среднего» размера $N_t \sim 20$, $N_x \sim 10$ Iloopt находит решение за ~ 1 сек., а SCIP работает в тысячу раз дольше. Только на поиск первого допустимого решения иногда уходит ~ 100 сек. И это для формулировки с SOS2. (На одном CPU машины Intel i7-6700 CPU @ 3.40GHz, 32Gb)

Сочетание SPWL + SOS2

Алгоритм Ветвей-и-Границ (BnB) часто можно ускорить за счет хорошего начального приближения (допустимого решения с «хорошим» значением критерия). Оказалось, что формулировка SPWL это позволяет!

После применения Ipopt к задаче SPW (сглаженная ломанная) мы быстро получаем решение

$$\left\{ \tilde{x}(t_k) (k=0:N_t), \tilde{F}_i (i=0:N_x) \right\}$$

На его основе строим допустимое решение для SOS2, решая несложные системы линейных уравнений, связанные последовательно вычисляемыми значениями

$$x(t_k) (k=0:N_t), x(t_0) = \tilde{x}(t_0).$$

Для автономного ОДУ 1-го порядка:

$$k \doteq 0, x(t_0) = \tilde{x}(t_0);$$

$k := k+1$ и решаем системы ЛУ для $i=0:N_x-1$

$$\begin{bmatrix} \frac{1}{\Delta t} & -\tilde{F}_i & -\tilde{F}_{i+1} \\ \frac{1}{2\Delta x} & -x_i & -x_{i+1} \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} x(t_{k+1}) \\ \lambda_i^{(k)} \\ \lambda_{i+1}^{(k)} \end{bmatrix} = \begin{bmatrix} \frac{x(t_k)}{\Delta t} \\ \frac{x_0}{\Delta x} - \frac{x(t_k)}{2\Delta x} \\ 1 \end{bmatrix}$$

пока не получим $\lambda_{i|i+1}^{(k)} \in [0, 1]$, зафиксируем $x(t_{k+1})$ и вернемся

Заключение и планы

- Сочетания различных способов дискретизации 1D-функций для представления 2D-функции, подлежащих идентификации в SvF-технологии, позволяет заметно сократить число неизвестных в получаемых ЗМП
- Приведённая методика применима к функциям трех и более (!) «фазовых» переменных
- Планируется апробация на демонстрационных примерах задач «небесной механики», где, например, «определяется» закон всемирного тяготения по зашумленным траекториям движения «планет» и естественно возникают ДУ с функциями 2 и 3 переменных в правой части
- Планируется применить параллельные реализации солвера глобальной и дискретной оптимизации SCIP (v9.2.1, v10) (**FiberSCIP**, ParaSCIP). *Только в этом году, разработчики параллельных реализаций устранили серьезные недостатки, не позволявшие полноценно использовать предыдущие версии*

Благодарю за внимание.

Вопросы?

<http://bit.ly/VoloshinovGScholar>

vladimir.voloshinov@gmail.com

Эксперименты и “внедрение” в SvF

- Тестовая реализация изложенных методов доступна здесь
<https://github.com/distcomp/SvF/test>
- Представление SPWL (сглаженная PWL-функция) уже успешно применялась в обратной задаче нелинейной теплопроводности (УРЧП)
- В экспериментах с комбинированной схемой расчётов SPWL+SOS2 наблюдалось многократное (3-5 раза) ускорение для размеров сеток ($N_t \sim 20$, $N_x \sim 10$). **2000 : 400** сек, **2000 : 600** сек и т.п., для примеров ДУ 1-и 2-го порядков,

$$[\dot{x}=x, x(t)=e^t], [\dot{x}=x^2, x(t)=\frac{1}{1-t}], [\ddot{x}=-4x, x(t)=\sin 2x + \cos 2x]$$

и точностью глобального решения обратной задачи (по критерию) 5%

- В рассмотренных примерах оказалось, что локальный оптимум для SPWL очень близок к «приближенно глобальному» в задаче SOS2 См. **Волошинов ВВ, Соколов АВ. Опт. методы и кусочно-линейные аппроксимации в обратных задачах с автономными ДУ // ОРВСЭУ-2022 в рамках НСКФ-2022, ИПС, 2023**