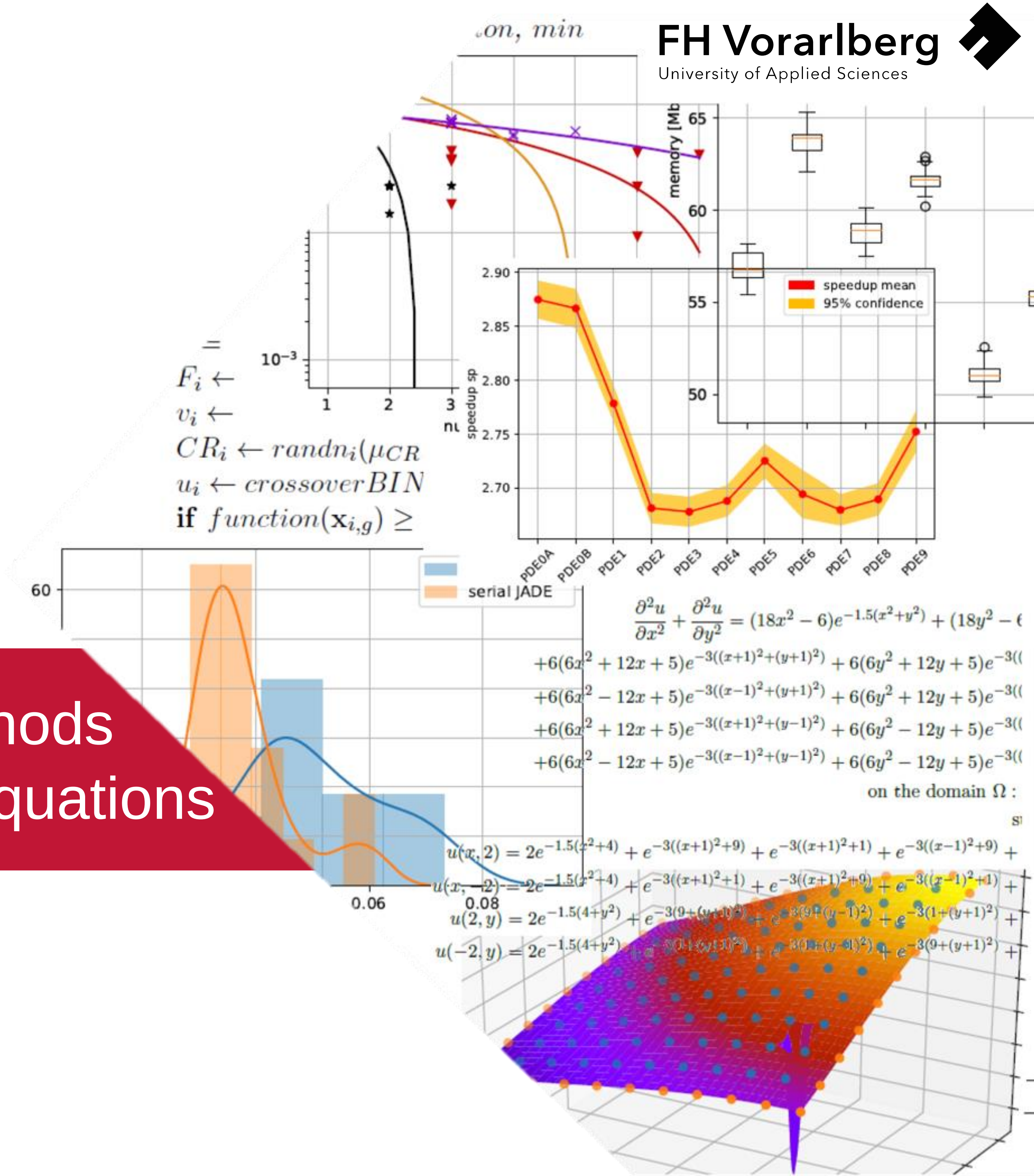
FH Vorarlberg
University of Applied Sciences

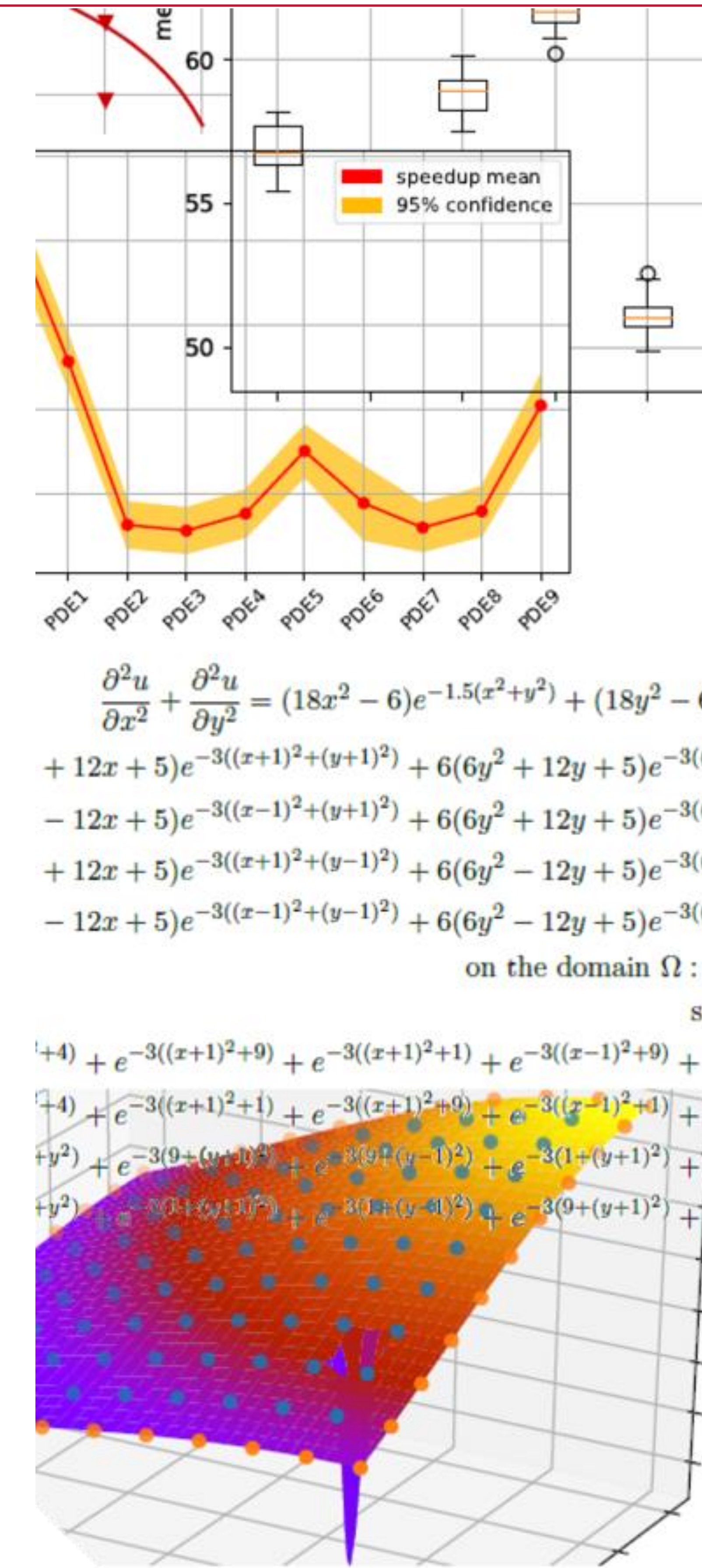


Computational Intelligence Methods for Solving Partial Differential Equations

Name: Nicolai Schwartze
Study Program: Master Mechatronics
Semester: 4

Outline

- Problem Description
- Experimental Design
- Serial JADE
- Parallel JADE
- Adaptive Kernel
- Gauss-Sine Kernel
- Conclusion



Problem Description: Optimisation Problem

- Reformulation of the differential equation into an optimisation problem
  - Minimisation of the residuum

  - x is limited to $x \in \mathbb{R}^2$
  - $\Omega \subseteq \mathbb{R}^2$ is the domain of the differential equation with the boundary $\partial\Omega$
  - $u(x)\colon \Omega \to \mathbb{R}$ is the solution to the differential equation
  - $\begin{aligned} Lu(x) &= f(x) \; in \; \Omega \\ Bu(x) &= g(x) \; auf \; \partial\Omega \end{aligned}$ L and B are linear differential operators

  - $R\big(u(x)\big) = Lu(x) - f(x)$
  - $R\big(u(x)\big) \to min$ for $u(x)$

Problem Description: Function Representation

- Approximation is represented as a finite sum of radial basis function

$$u(x) \approx u_{apx}(x) = \sum_{i=0}^{N} \phi_i(x)$$

- Two basis functions are tested
  - Gauss Kernel

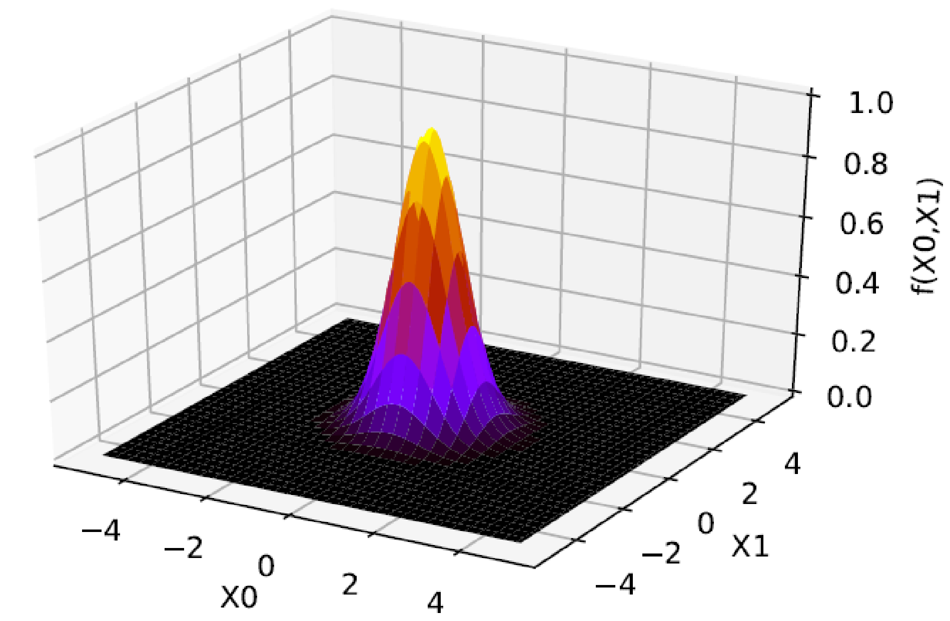
$$gak(x) = \omega e^{-\gamma \|x-c\|^2}$$

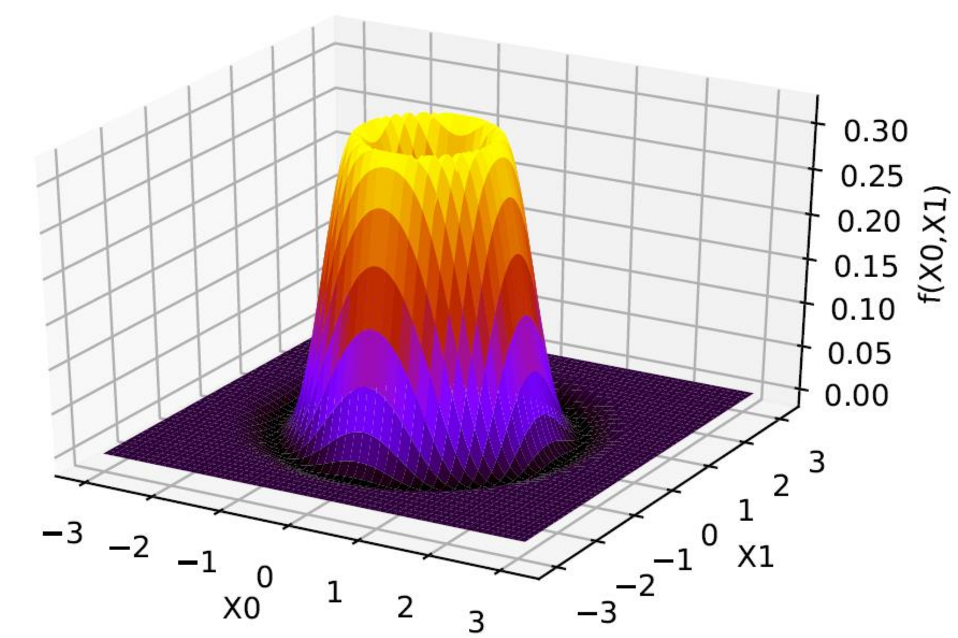
$$[\omega_0, \gamma_0, c_{00}, c_{01}, \dots\ \omega_i, \gamma_i, c_{i0}, c_{i1}, \dots\ \omega_N, \gamma_N, c_{N0}, c_{N1}]^T$$

  - Gauss-Sine Kernel

$$gsk(x) = \omega e^{-\gamma \|x-c\|^2} \sin(f\|x-c\|^2 - \varphi)$$

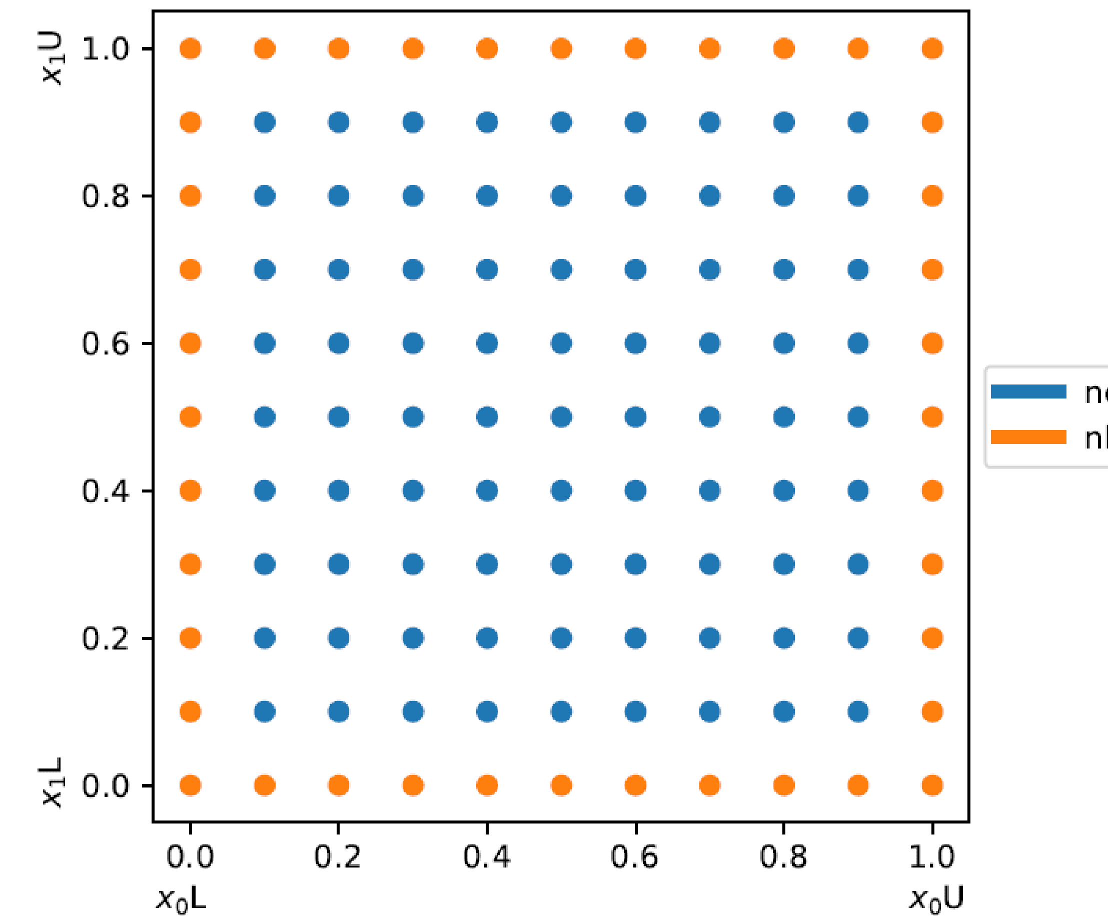
$$[\omega_0, \gamma_0, c_{00}, c_{01}, f_0, \varphi_0, \dots\ \omega_i, \gamma_i, c_{i0}, c_{i1}, f_i, \varphi_i, \dots\ \omega_N, \gamma_N, c_{N0}, c_{N1}, f_N, \varphi_N]^T$$





Problembeschreibung: Fitnessfunktion

- Based on the residuum
- Assigns a real value to an approximation $u_{apx}(x)$
- Residuum for $u_{apx}(x)$ is evaluated
  - Finite number of Collocation Points in the domain
- „Using CMAES for solving different types of differential equations“ (Chaquet und Carmona 2019)



- Fitness function:

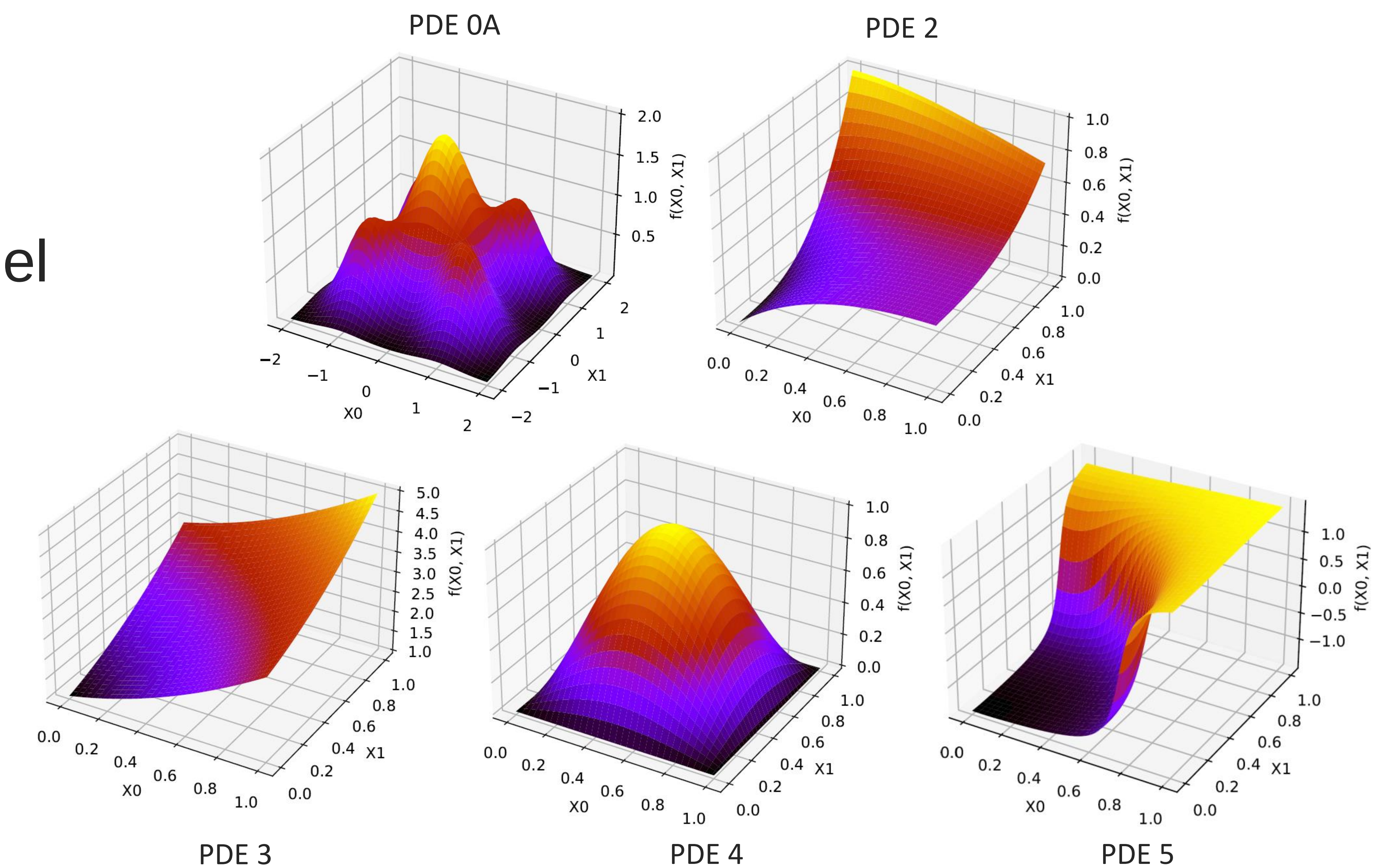
$$F\left(u_{apx}(x)\right) = \frac{\sum_{i=1}^{nc} \xi(x_i)\left[Lu_{apx}(x_i) - f(x_i)\right]^2 + \phi \sum_{j=1}^{nB}\left[Bu_{apx}(x_j) - g(x_j)\right]^2}{(n_C + n_B)}$$

- Implementation: $F\left(\underbrace{u_{apx}(x)}_{[\,\omega_i, \gamma_i, c_{i0}, c_{i1}]^{\;T}}\right) : \mathbb{R}^{4N} \to \mathbb{R}$

Experimental Design: Testbed

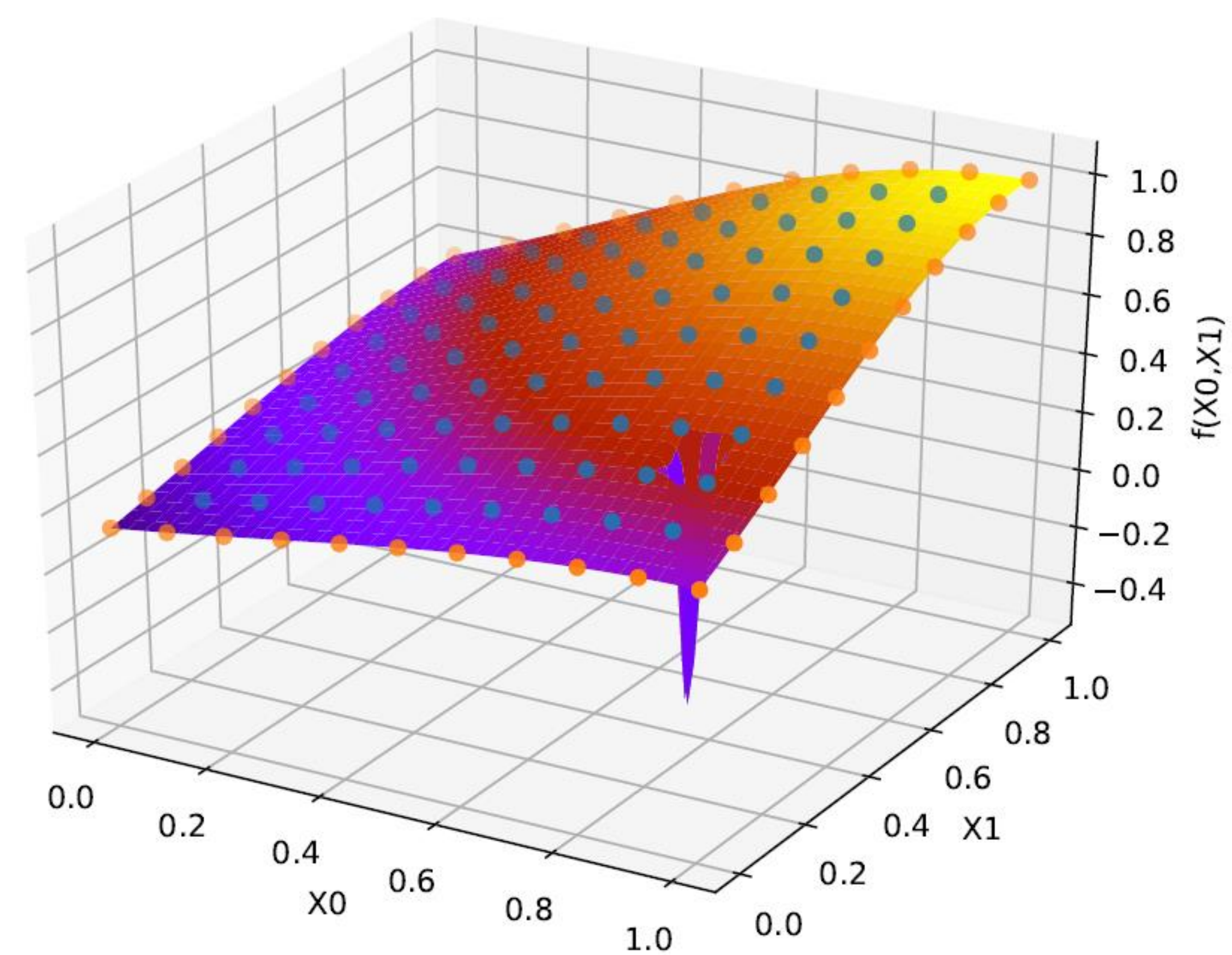
- Testbed consists of 11 PDEs
  - Poisson Equation:
    Laplace operator is applied to solution function

- 5 Equations are important:
  - PDE 0A: Sum of 5 Gauss Kernel
  - PDE 2: From Literature
  - PDE 3: From Literature
  - PDE 4: From NIST Testbed
  - PDE 5: From NIST Testbed



Experimental Design: Comparison

- 3 attributes of the solver are compared:
  - Execution time
  - Allocated memory
  - Quality of the approximation
- Compared to Finite Element Solver NGSolve
- Compared to other results from the literature
- Statistical interpretation: 20 replications
  Wilcoxon Ranksum Test with α = 0.05

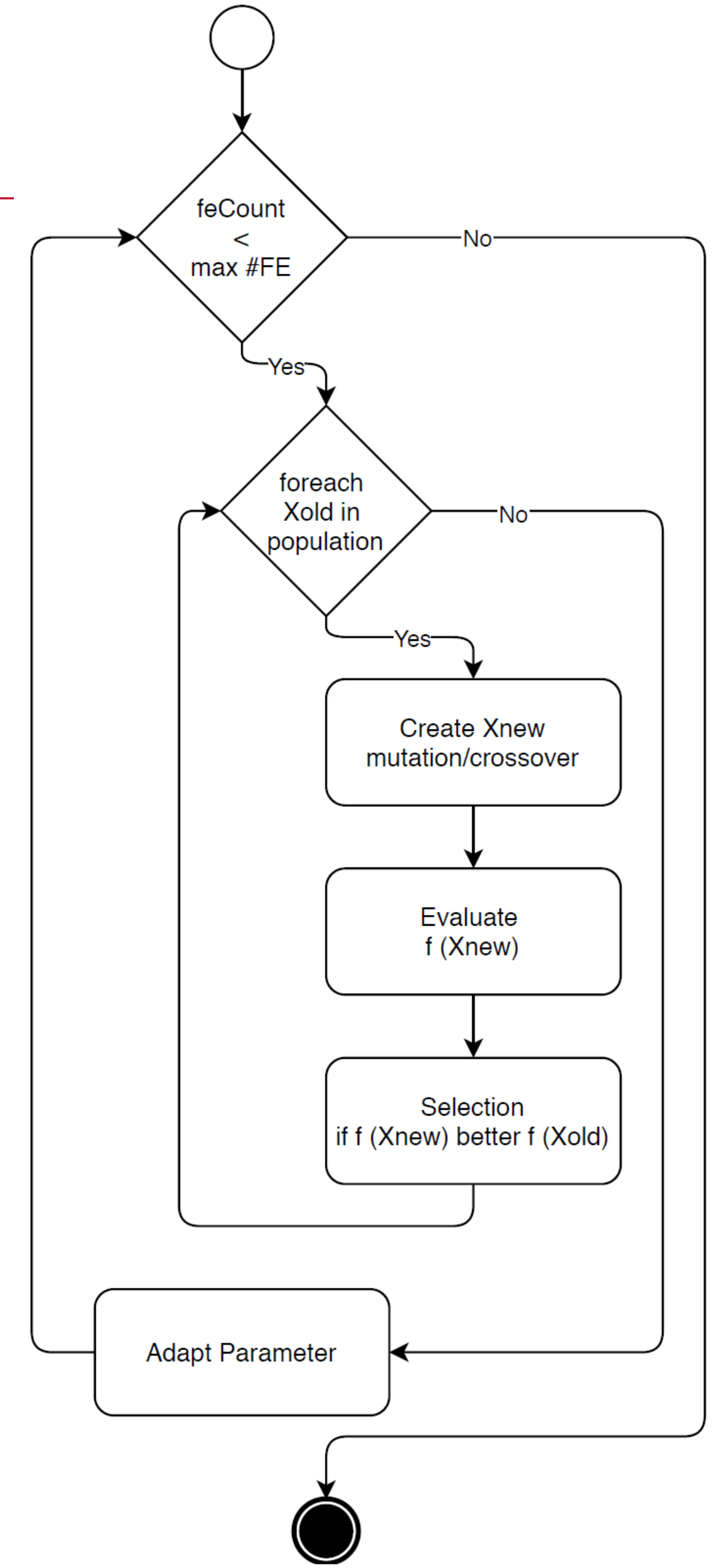


$$\left\| u_{ext} - u_{apx} \right\| = \sqrt{\int_{\Omega} \left(u_{ext}(x) - u_{apx}(x) \right)^2 dx}$$

Serial JADE: Hypothesis

- JADE is an evolutionary optimisation algorithm
  - Mutation/Crossover
  - Evaluation
  - Selection
- The same experiments as in „Chaquet and Carmona 2019“
  - JADE instead of CMAES
  - Slight different parameter

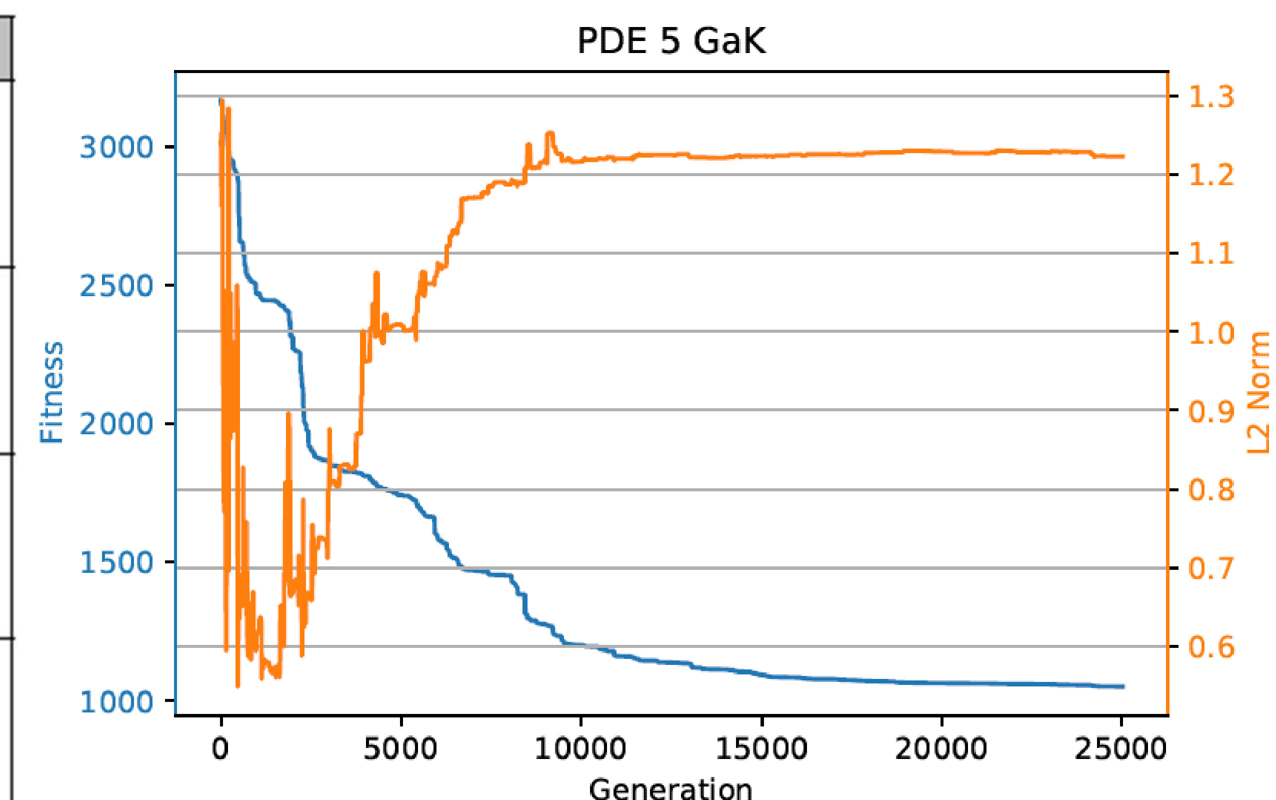
- Is it possible to use JADE and achieve comparable results to other algorithms from the literature research?



Serial JADE: Results

Name	10^4 #FE		10^6 #FE		Wilcoxon
	mean	median	mean	median	
PDE 0A	1.9415 ± 0.3321	1.8844	0.6596 ± 0.5510	0.9285	sig. better
PDE 2	0.0890 ± 0.0334	0.0760	0.0257 ± 0.0140	0.0224	sig. better
PDE 3	0.2409 ± 0.1051	0.2309	0.0328 ± 0.0169	0.0285	sig. better
PDE 4	0.1102 ± 0.0367	0.0985	0.0378 ± 0.0083	0.0352	sig. better
PDE 5	0.6645 ± 0.1930	0.6263	1.1968 ± 0.0286	1.2056	sig. worse

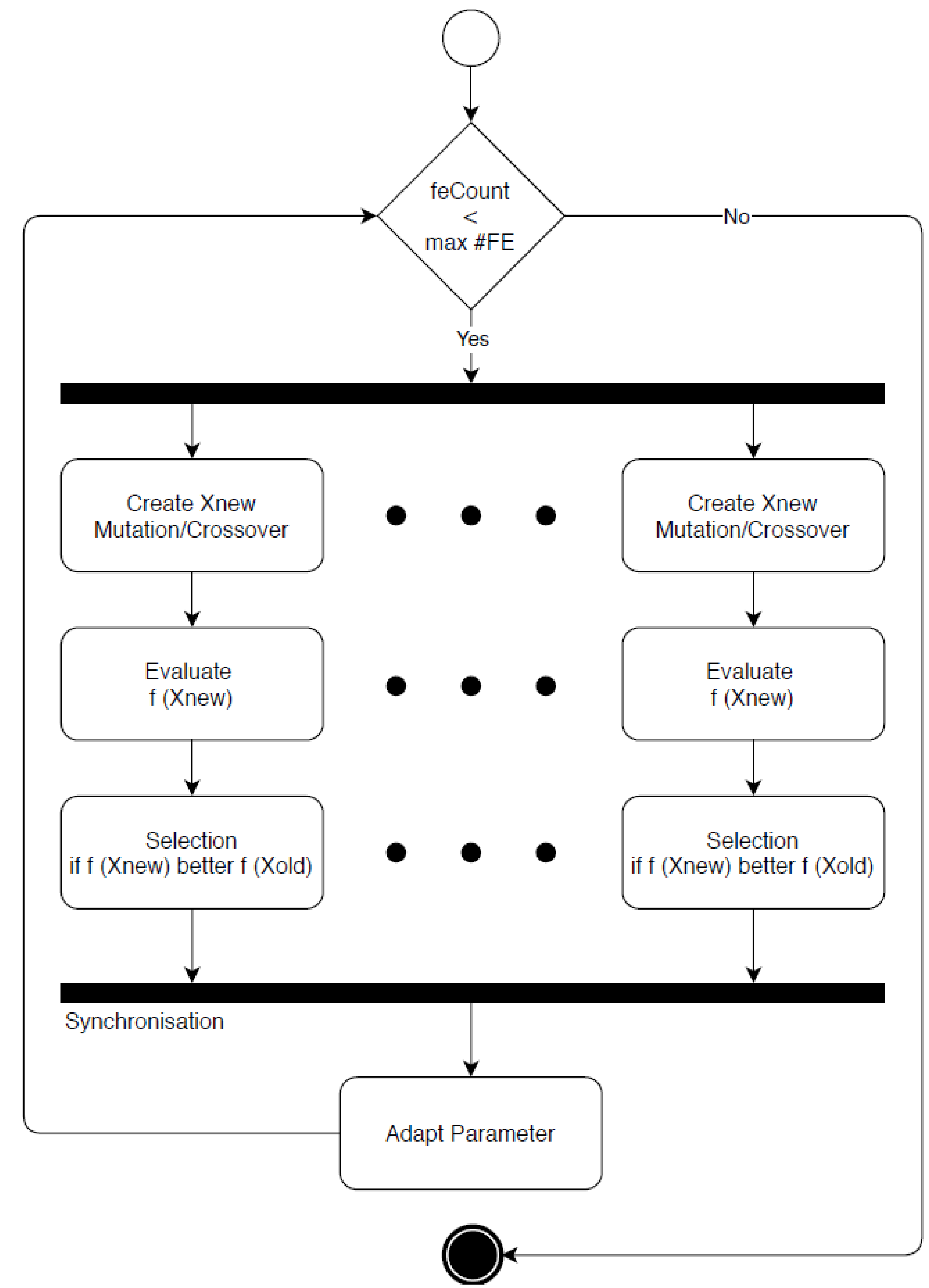
Paper	Parameter	RMSE PDE 2	RMSE PDE 3
Chaquet and Carmona 2019	4 kernel max #FE=10^6 50 replications	$(1.75 \pm 1.14)10^{-4}$	$(1.09 \pm 0.846)10^{-5}$
Chaquet and Carmona 2012	10 harmonics max #FE $= G \cdot \lambda = 1.2 \cdot 10^6$ 10 replications	$(6.37 \pm 0.733)10^{-3}$	$(5.90 \pm 0.799)10^{-3}$
Panagant and Bureerat 2014	unknowns: N/A #FE=$5 \cdot 10^5$ replications: N/A	$7.256 10^{-4}$	$9.489 10^{-6}$
serial memetic JADE	5 kernel max #FE $= 10^6$ 20 replications	$(2.9798 \pm 1.5541)10^{-2}$	$(3.8225 \pm 1.9438)10^{-2}$



Parallel JADE: Hypothesis

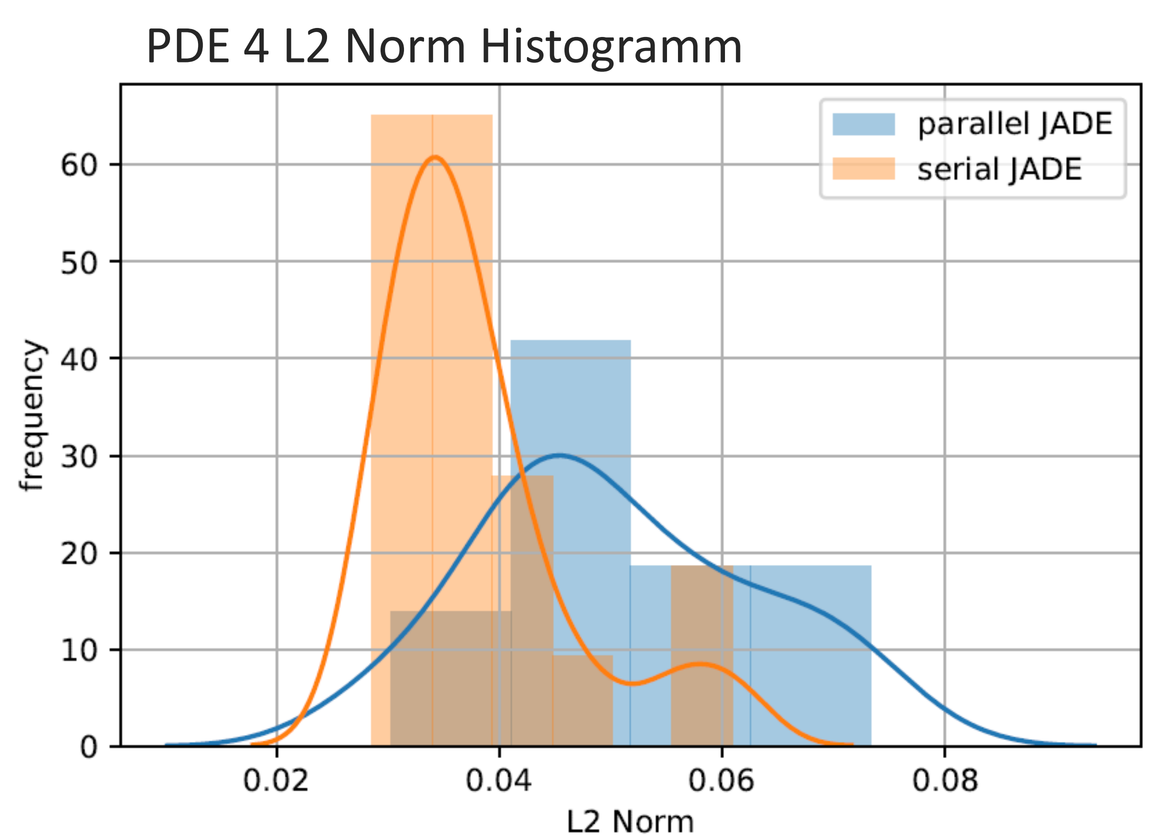
- Parallel implementation of JADE
- The inner iteration over the population is evaluated in parallel
- From an algorithm perspective only slightly different:
  - The information is only available after the synchronisation of the processes

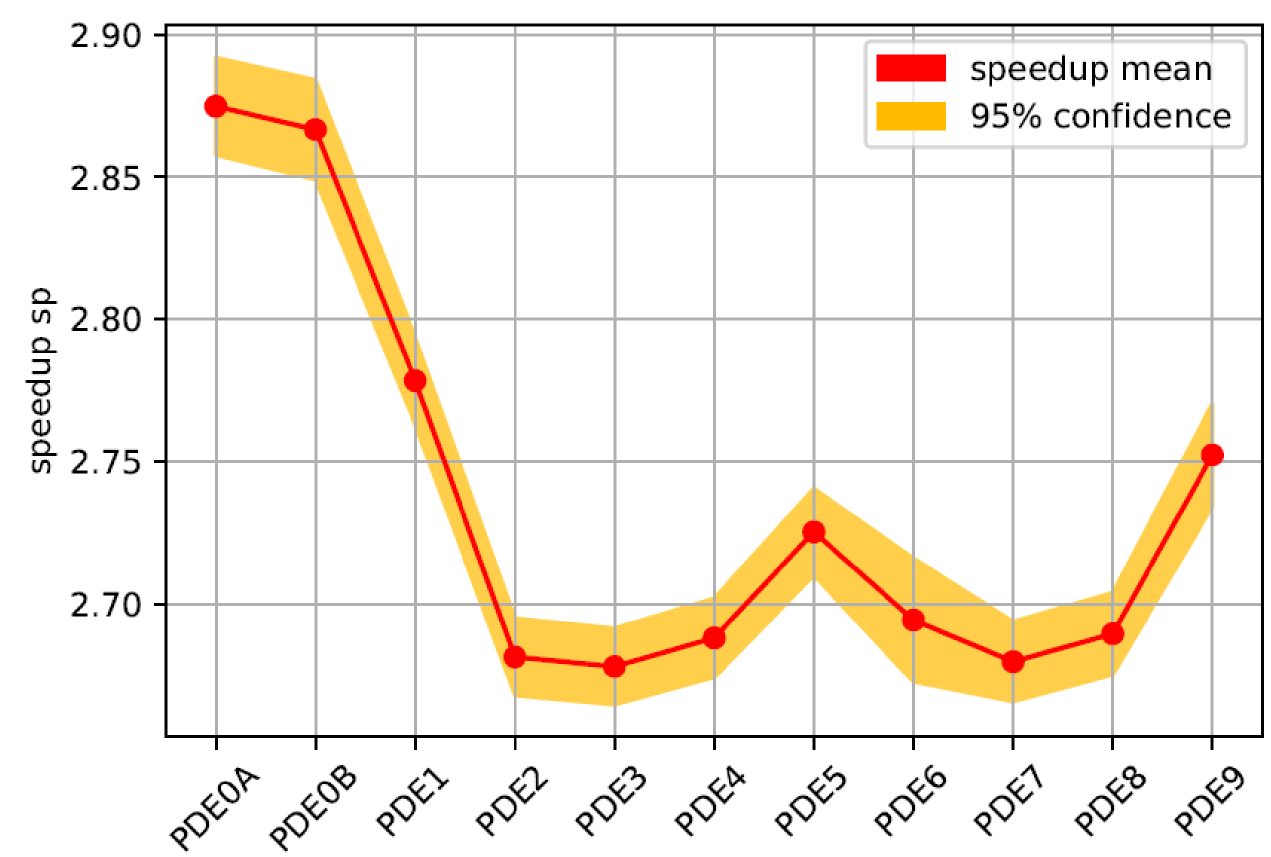
- Can the parallel algorithm decrease the execution time of the program?



Parallel JADE: Results

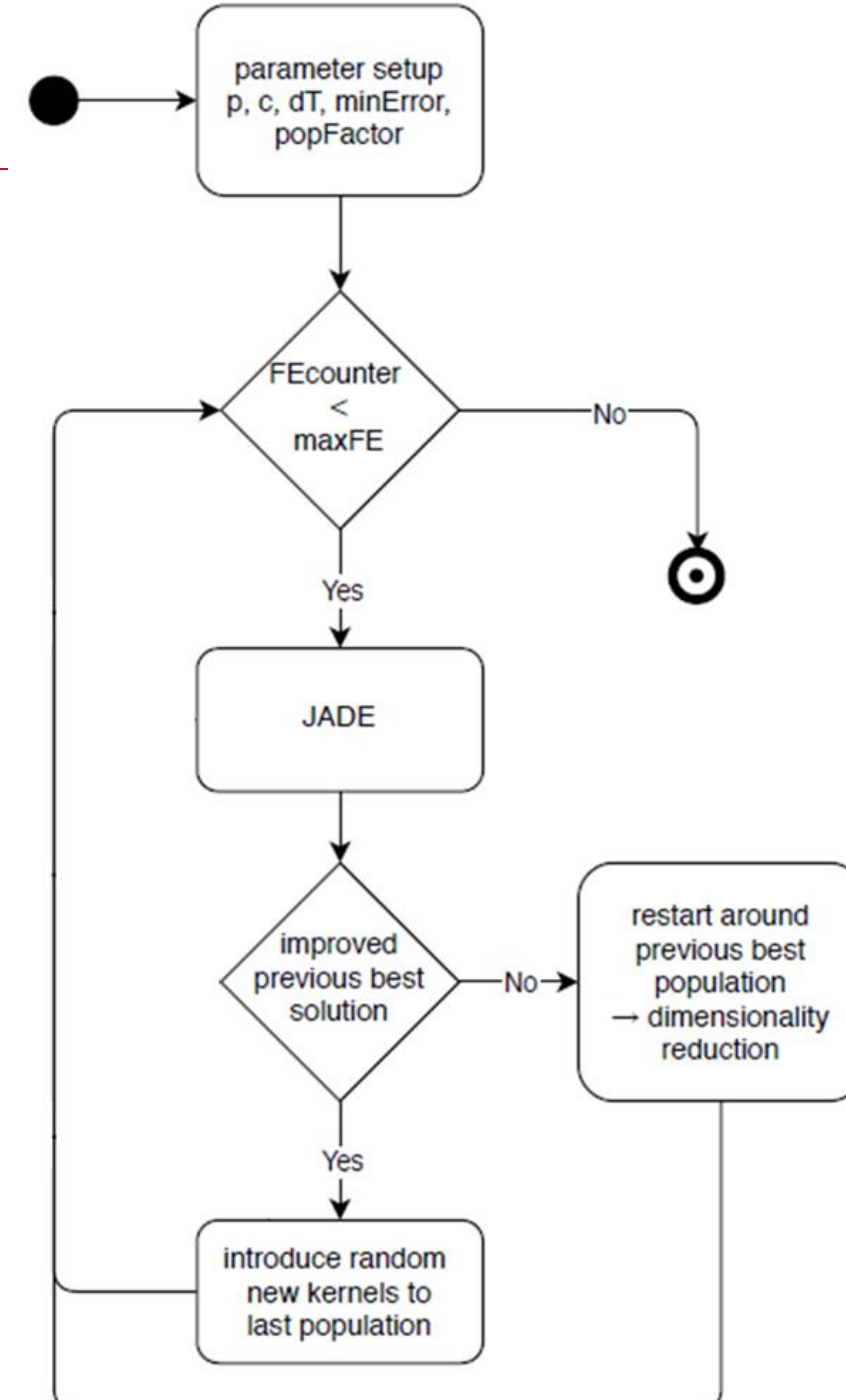
Name	serial JADE		parallel JADE		Wilcoxon
	mean	median	mean	median	
PDE 0A	0.6596 ± 0.5510	0.9285	0.6939 ± 0.6635	0.9243	unsig. undecided
PDE 2	0.0257 ± 0.0140	0.0224	0.0300 ± 0.0157	0.0255	unsig. worse
PDE 3	0.0328 ± 0.0169	0.0285	0.0371 ± 0.0206	0.0295	unsig. worse
PDE 4	0.0378 ± 0.0083	0.0352	0.0505 ± 0.0121	0.0481	sig. worse
PDE 5	1.1968 ± 0.0286	1.2056	1.2030 ± 0.0465	1.2053	unsig. undecided





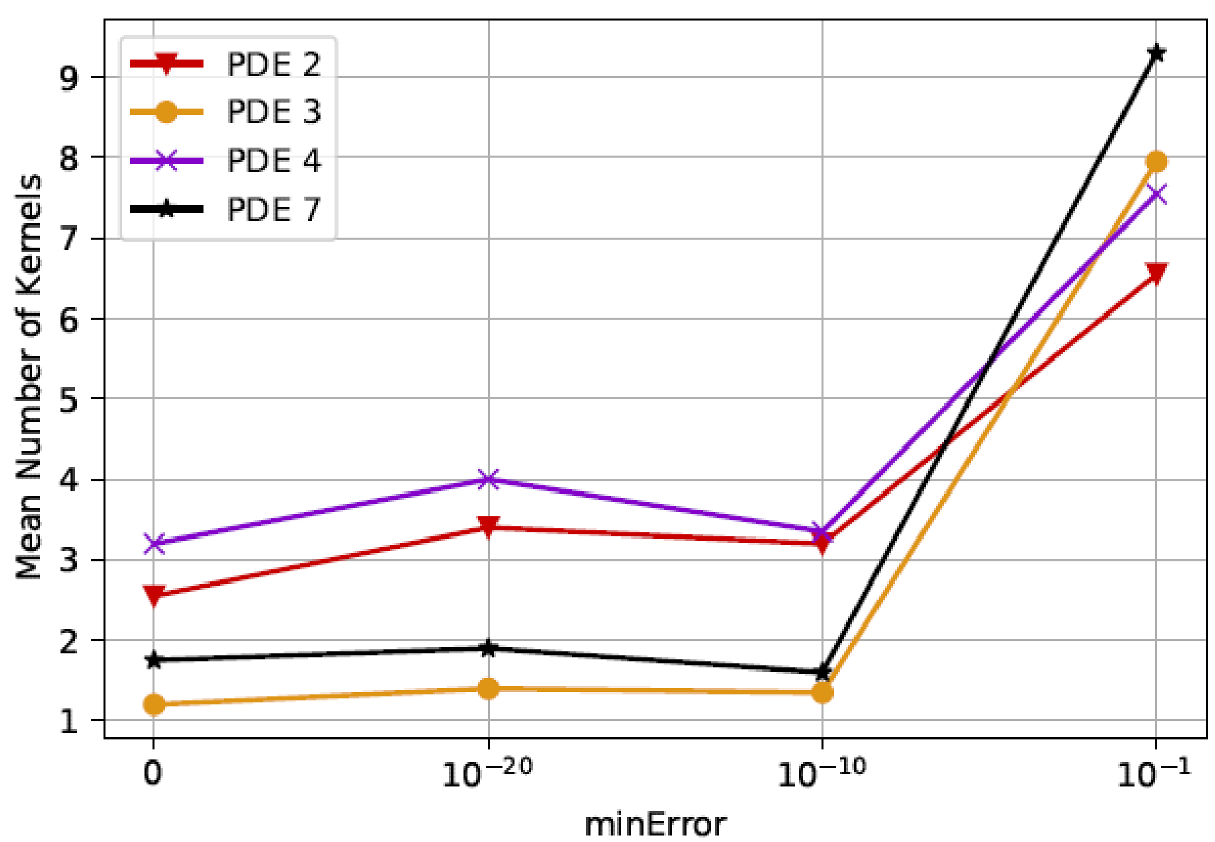
Adaptive Kernel: Hypothesis

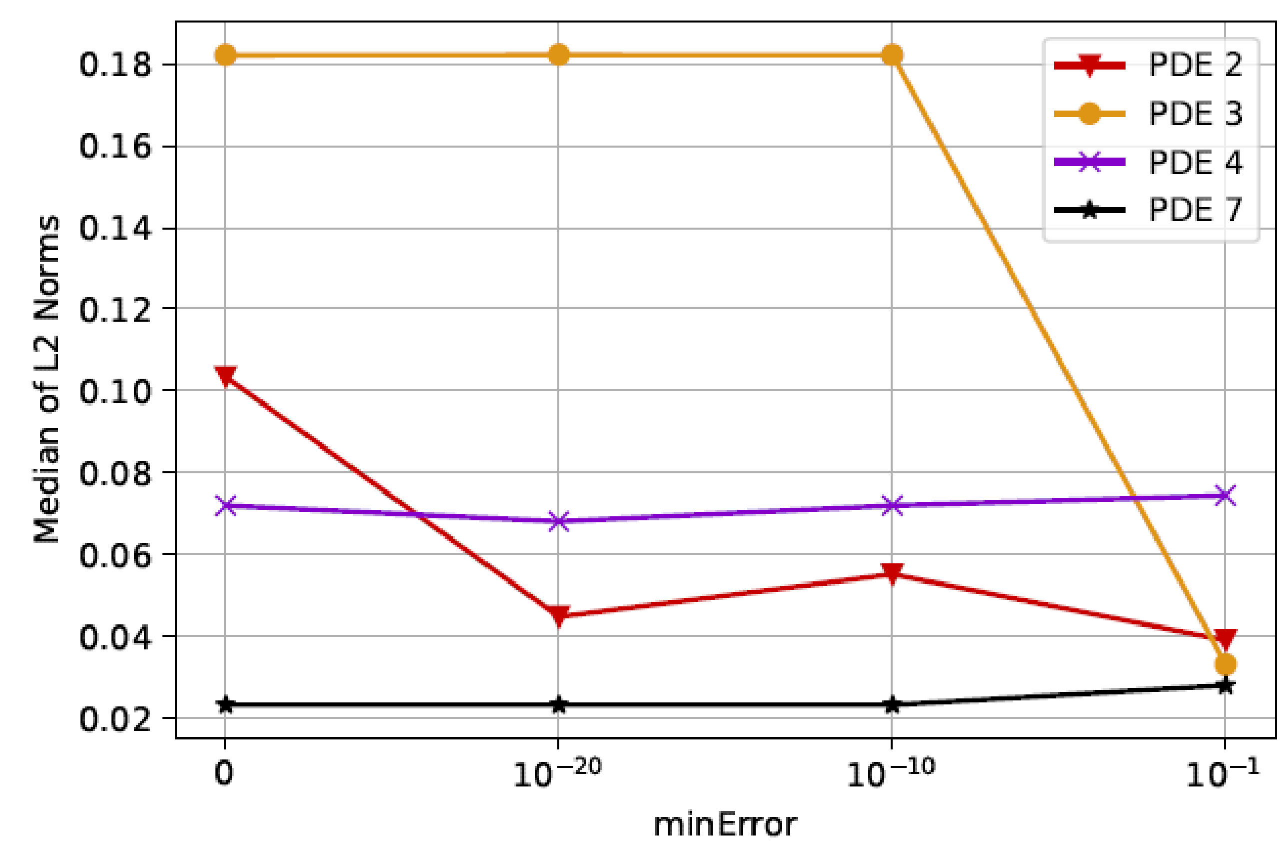
- Convergence based termination of JADE necessary:
  - Unchanged fitness over generations
- Adaptive Scheme
  - Start with one kernel, low dimension
  - Increase kernel count until the solution can not be improved with more kernel
- Can the adaptive scheme improve the results?



Adaptive Kernel: Results

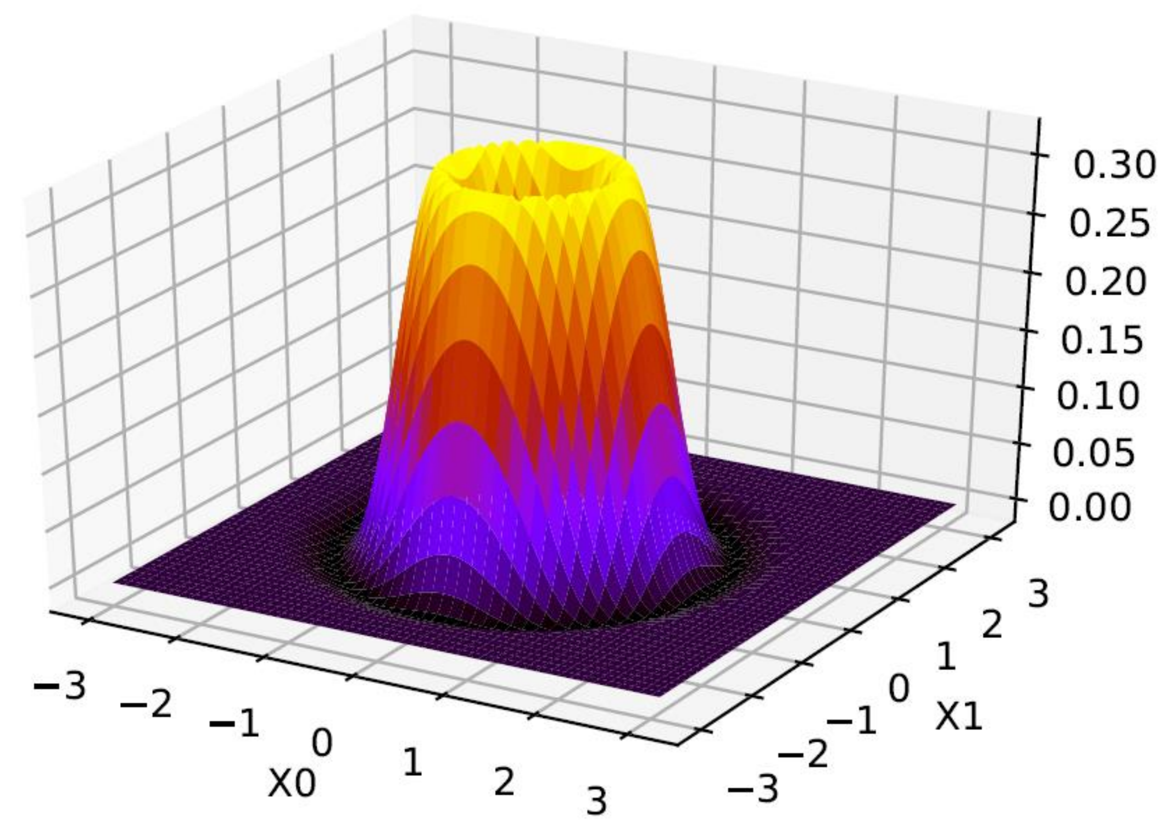
Name	parallel JADE		adaptive Kernel		Wilcoxon
	mean	median	mean	median	
PDE 0A	0.6939 ± 0.6635	0.9243	9.694E-16 ± 1.486E-16	9.255E-16	sig. better
PDE 2	0.0300 ± 0.0157	0.0255	0.0735 ± 0.0358	0.1034	sig. worse
PDE 3	0.0371 ± 0.0206	0.0295	0.1731 ± 0.0395	0.1822	sig. worse
PDE 4	0.0505 ± 0.0121	0.0481	0.0707 ± 0.0053	0.0720	sig. worse
PDE 5	1.2030 ± 0.0465	1.2053	122.6312 ± 372.5676	1.1643	unsig. undecided





Gauss-Sine Kernel: Hypothesis

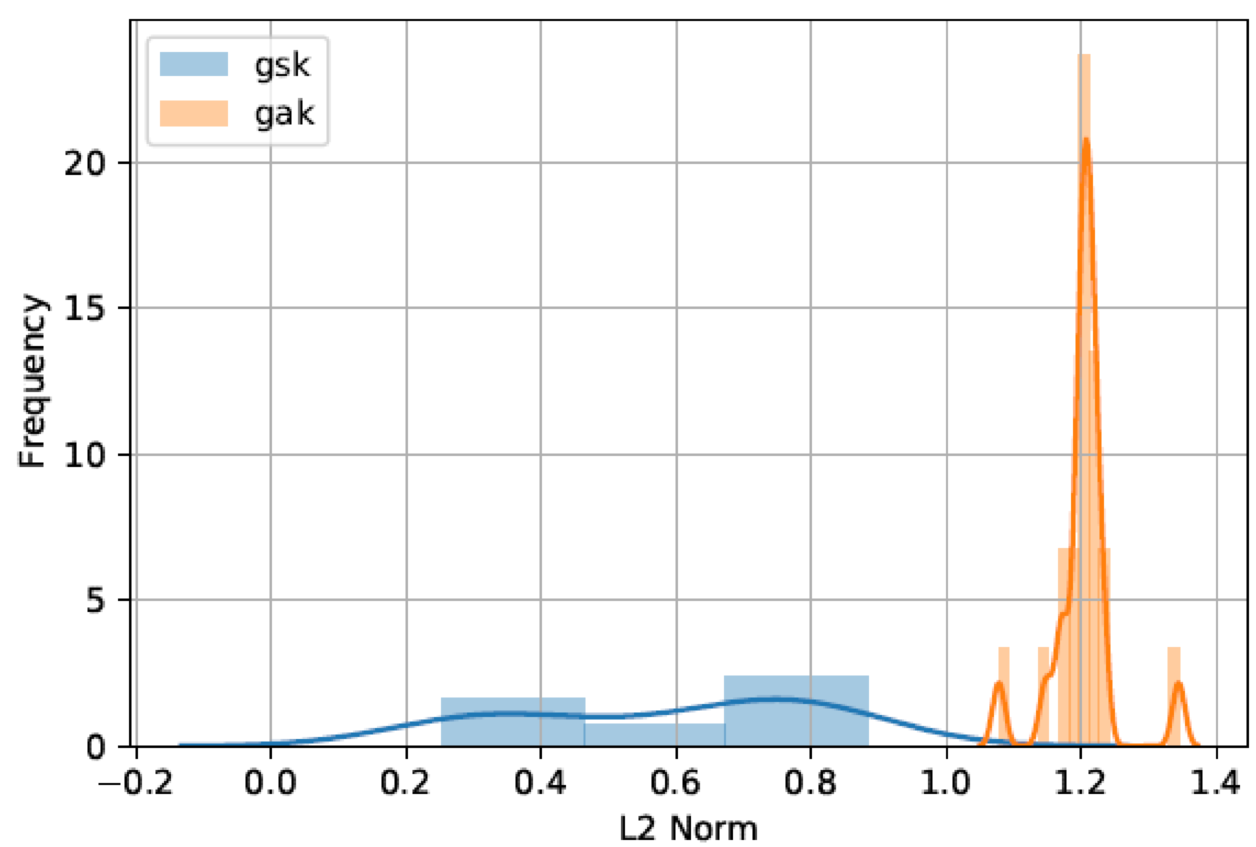
- Fitness function gets optimised but does not reflect the actual quality criterion
  - Fitness function must be changed
    $\rightarrow$ simple way to do that: other kernel
- $gsk(x) = \omega e^{-\gamma\|x-c\|^2} \sin(f\|x-c\|^2 - \varphi)$

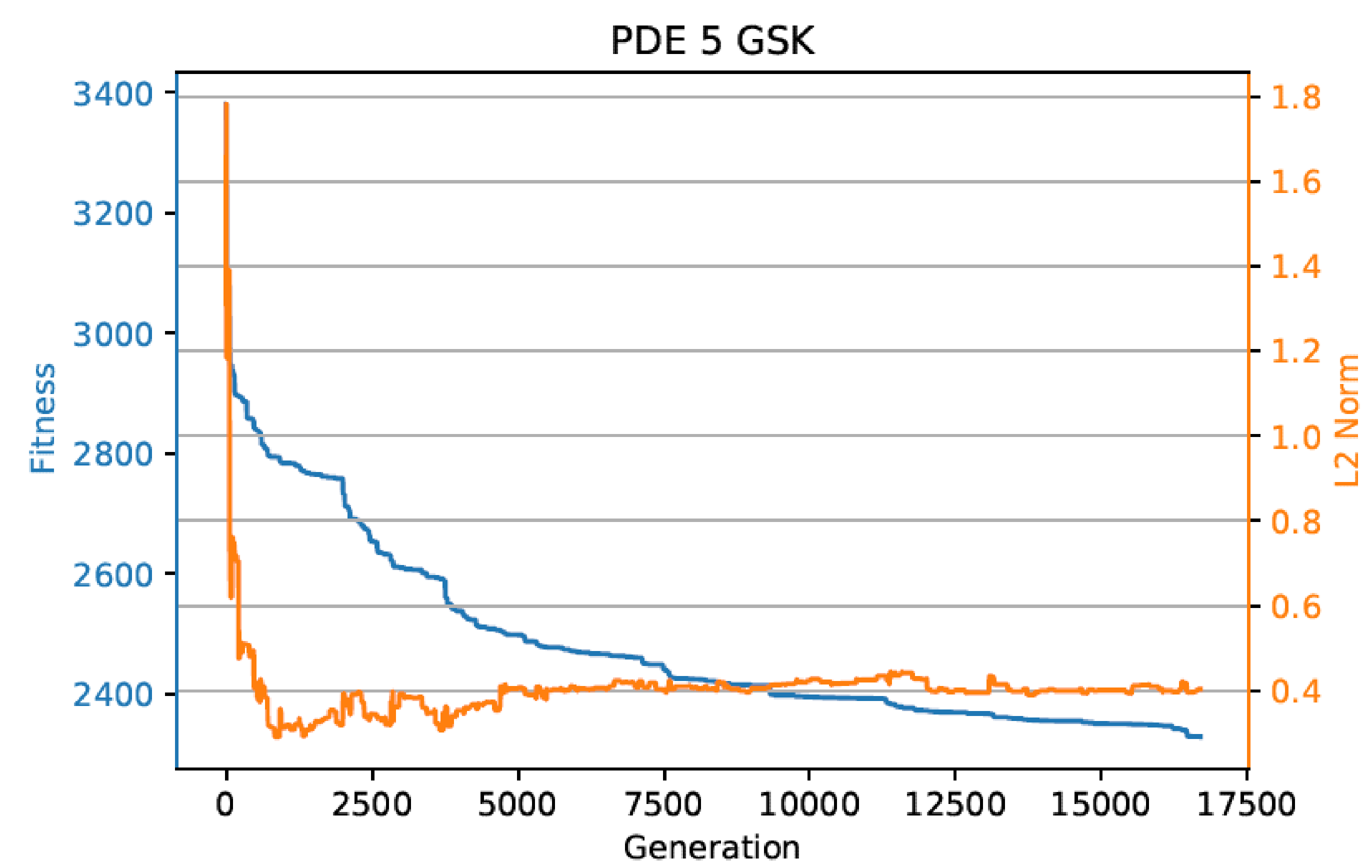


- Can the new Gauss-Sine Kernel cope with the problem of contradictory fitness and L2 Norm on PDE 5?

Gauss-Sine Kernel: Results

Name	Gauss Kernel		Gauss-Sine Kernel		Wilcoxon
	mean	median	mean	median	
PDE 0A	0.6939 ± 0.6635	0.9243	0.8106 ± 0.7929	0.6765	unsig. undecided
PDE 2	0.0300 ± 0.0157	0.0255	0.0448 ± 0.0224	0.0416	unsig. worse
PDE 3	0.0371 ± 0.0206	0.0295	0.0263 ± 0.0111	0.0269	unsig. better
PDE 4	0.0505 ± 0.0121	0.0481	0.0470 ± 0.0078	0.0458	unsig. better
PDE 5	1.2030 ± 0.0465	1.2053	0.5860 ± 0.2149	0.6841	sig. better





Conclusion

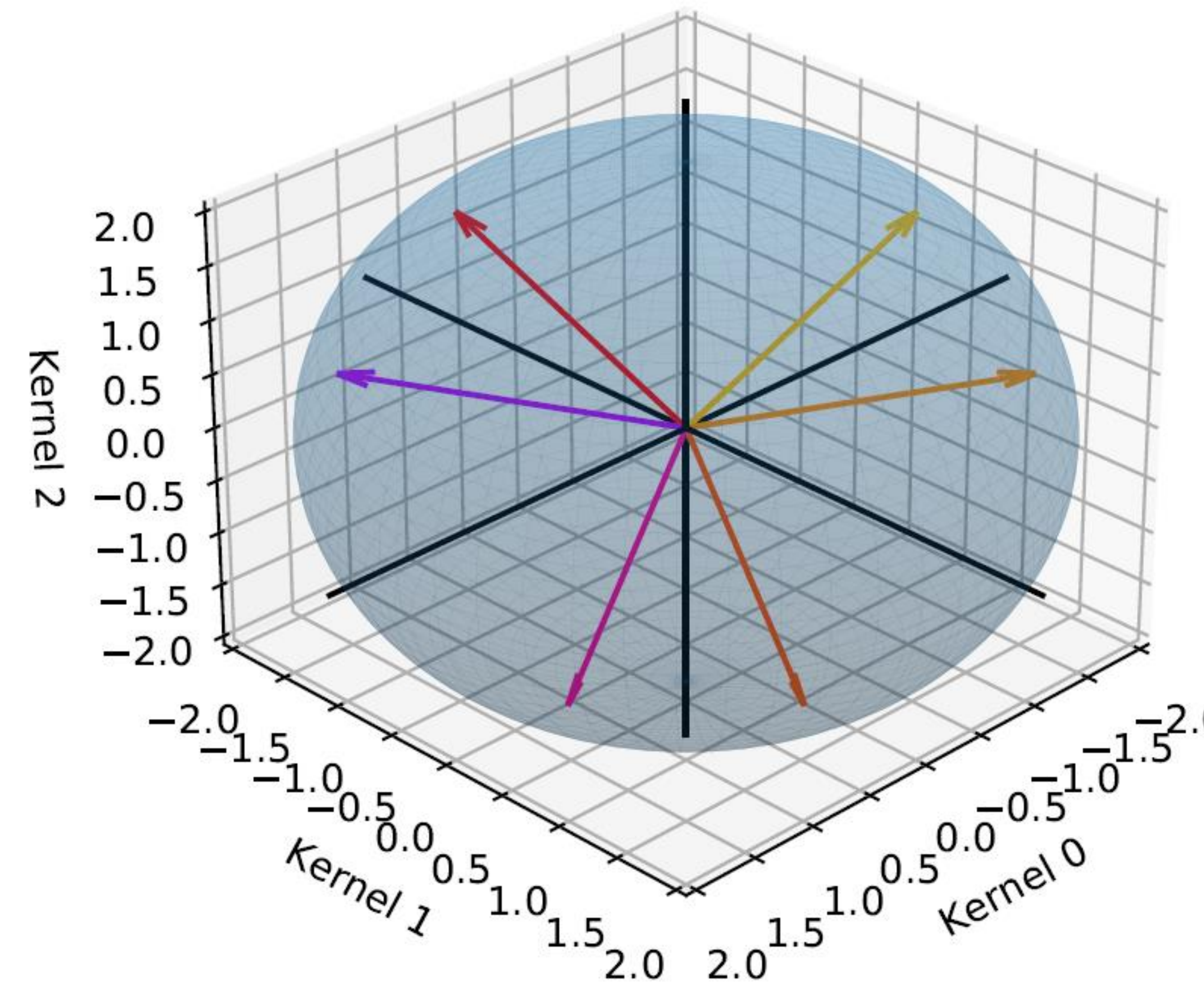
- JADE produces worse results then other comparable algorithms from the literature

- Use CMAES on this testbed

- A radial symmetry of the fitness function can be observed, which could lead to the design of better suited algorithms

$$u(x) \approx u_{apx}(x) = \sum_{i=0}^{N} \phi_i(x)$$

$$\left[\underrightarrow{kernel_0}, \dots \underleftrightarrow{kernel_i}, \dots \underleftarrow{kerenl_N}\right]^T$$



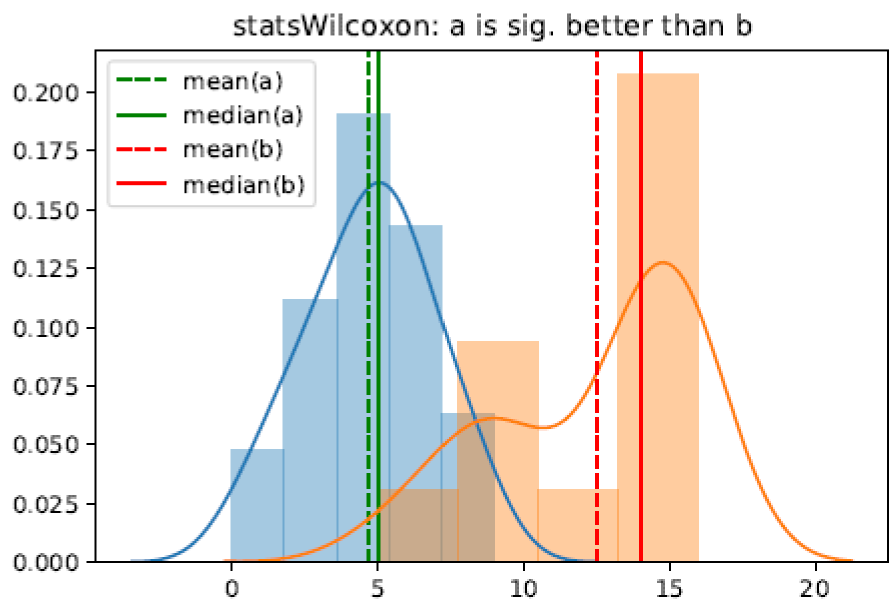
FH Vorarlberg
University of Applied Sciences

www.fhv.at

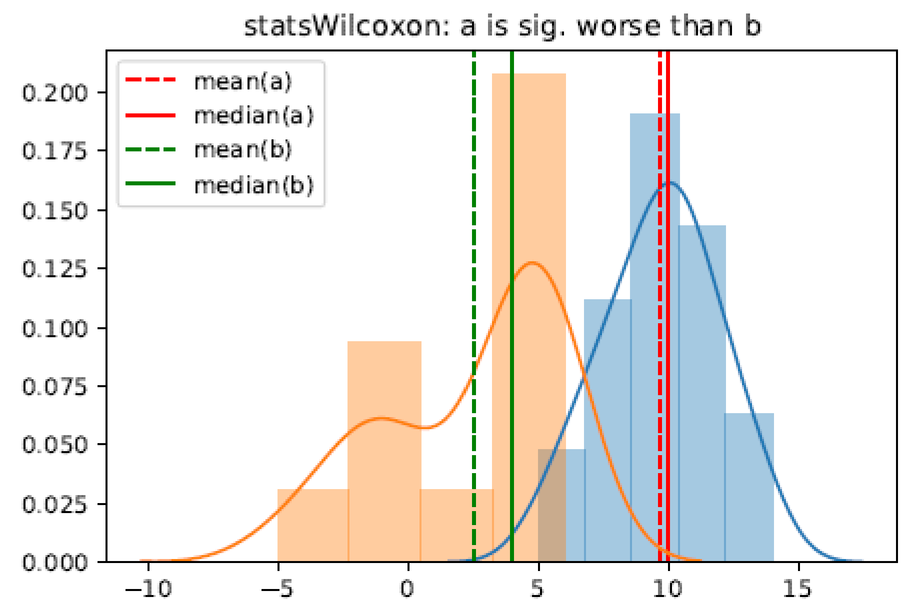
Name: Nicolai Schwartze
Contact: nicolai.schwartze@students.fhv.at

Study Program, Semester: Master Mechatronics, 4

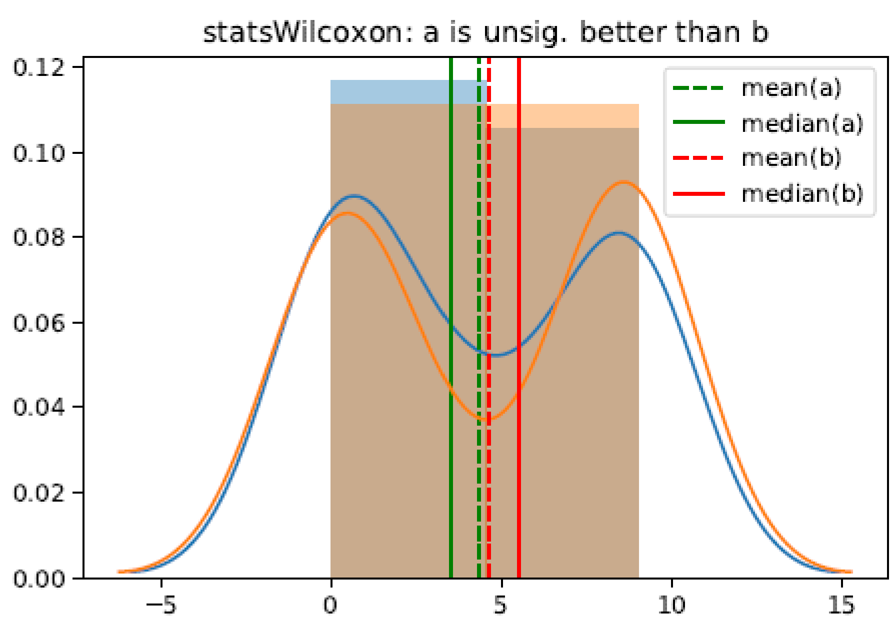
Statistical Significance: Wilcoxon Test



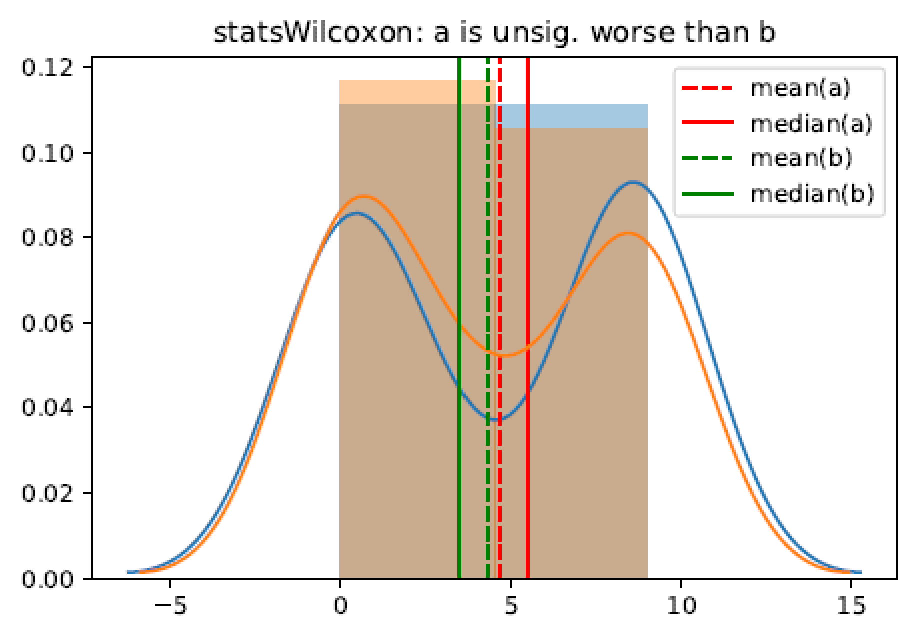
(a) A is significantly better than B



(b) A is sigificantly worse than B

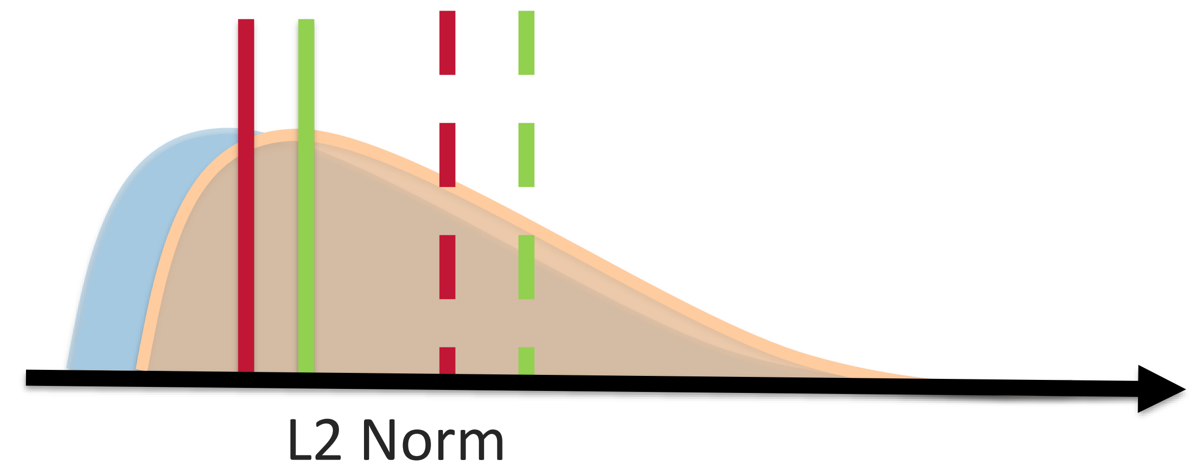


(c) A is unsignificantly better than B



(d) A is unsignificatnly worse than B

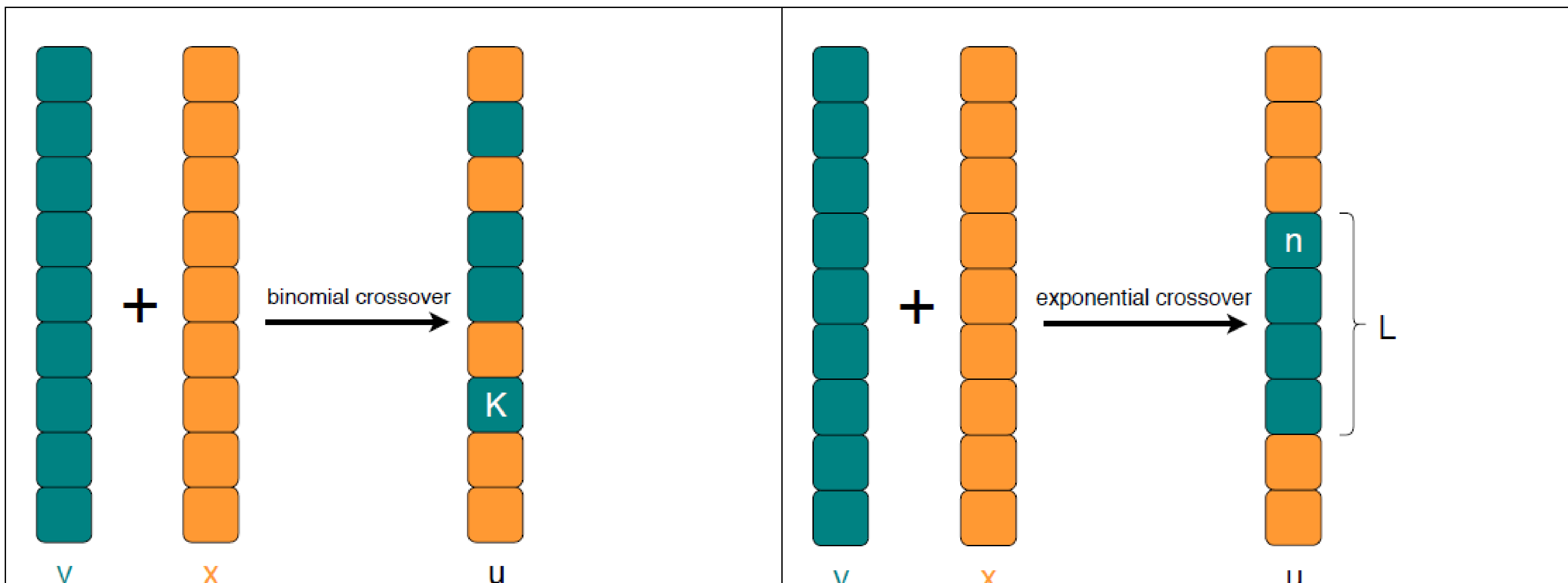
Undecided:



Pseudocode: Mutation/Crossover

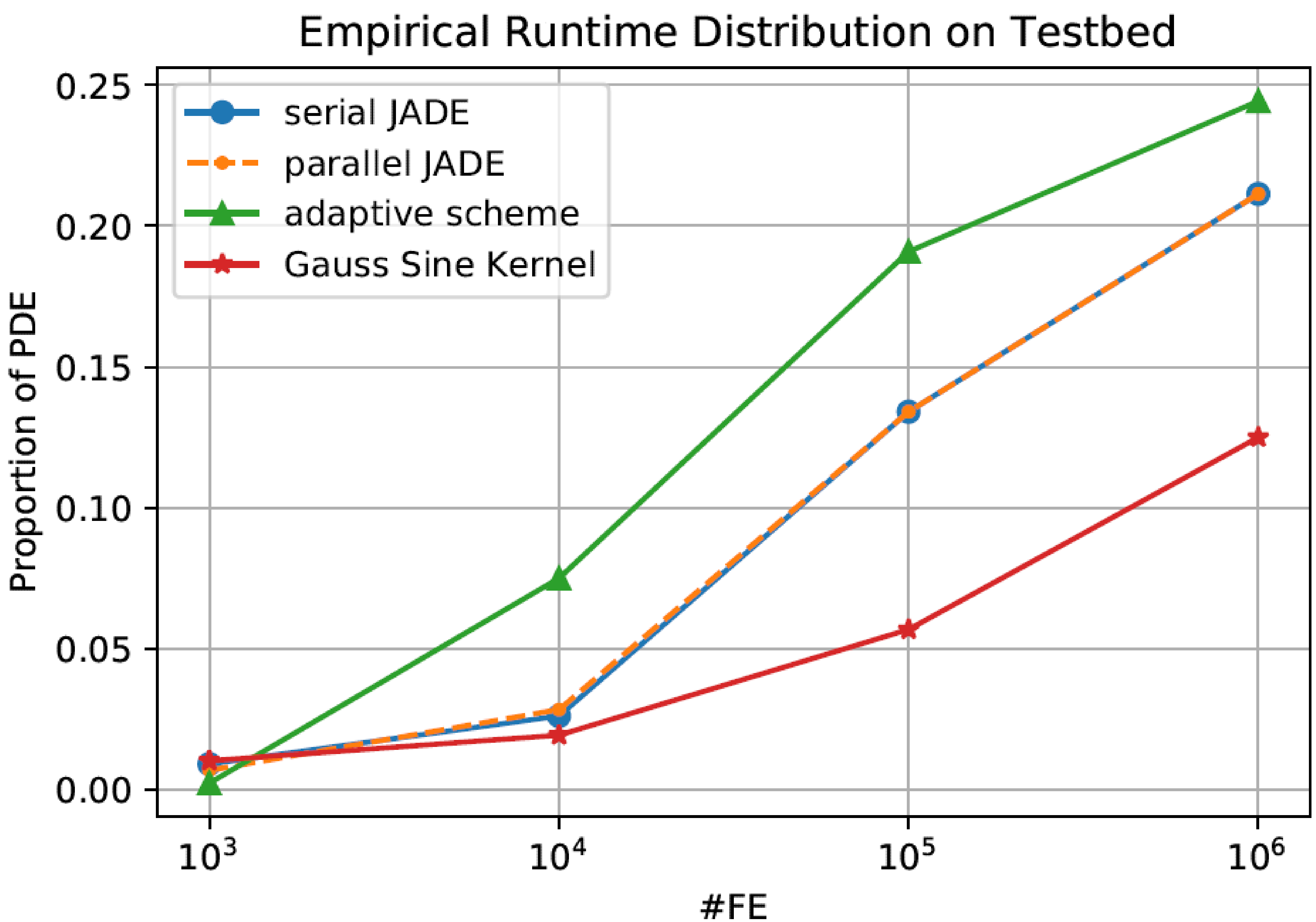
CurrentToPBest	$v_{i,g} = x_{i,g} + F_i \cdot \left(x^p_{best,g} - x_{i,g}\right) + F_i \cdot \left(x_{r1,g} - x_{r2,g}\right)$
CurrentToPBest with archive	$v_{i,g} = x_{i,g} + F_i \cdot \left(x^P_{best,g} - x_{i,g}\right) + F_i \cdot \left(x_{r1,g} - \tilde{x}_{r2,g}\right)$

binomial crossover	exponential crossover



$u_j = \begin{cases} v_j, & \text{if } j = K \vee rand[0,1] \leq CR \\ x_j, & \text{otherwise} \end{cases}$	$u_j = \begin{cases} v_j, & \text{if } j \in \{\langle n \rangle_d, \ldots \langle n + L - 1 \rangle_d\} \\ x_j, & \text{otherwise} \end{cases}$

Expected Runtime Distribution



L2 Norm auf allen 11 PDEs zu einem Targetvalue von:

0.05

0.01

0.005

0.001

Pseudocode: JADE

Algorithm A.1: JADE Pseudocode

Function `JADE`($\mathbf{X}_{g=0}$, p, c, $function$, $minError$, $maxFE$):
 $fValue_{g=0} \leftarrow function(\mathbf{x}_{g=0})$
 $\mu_{CR} \leftarrow 0.5$
 $\mu_F \leftarrow 0.5$
 $A \leftarrow \emptyset$
 while $fe \leq maxFE$ **do**
  $g \leftarrow g + 1$
  $S_F \leftarrow \emptyset$
  $S_{CR} \leftarrow \emptyset$
  for $i = 1$ *to* NP **do**
   $F_i \leftarrow randc_i(\mu_F, 0.1)$
   $v_i \leftarrow mutationCurrentToPBest1(\mathbf{x}_{i,g}, A, fValue_g, F_i, p)$
   $CR_i \leftarrow randn_i(\mu_{CR}, 0.1)$
   $u_i \leftarrow crossoverBIN(\mathbf{x}_{i,g}, v_i, CR_i)$
   if $function(\mathbf{x}_{i,g}) \geq function(\mathbf{u}_{i,g})$ **then**
    $\mathbf{x}_{i,g+1} \leftarrow \mathbf{x}_{i,g}$
   end
   else
    $\mathbf{x}_{i,g+1} \leftarrow \mathbf{u}_{i,g}$
    $fValue_{i,g+1} \leftarrow function(\mathbf{u}_{i,g})$
    $\mathbf{x}_{i,g} \rightarrow \mathbf{A}$
    $CR_i \rightarrow S_{CR}$
    $F_i \rightarrow S_F$
   end
  end
  `// resize` A `to size of` $\mathbf{x}_g$
  if $|A| > NP$ **then**
   $A \leftarrow A \setminus A_{rand_i}$
  end
  $fe \leftarrow fe + size(\mathbf{X})$
  $\mu_{CR} \leftarrow (1 - c) \cdot \mu_{CR} + c \cdot arithmeticMean(S_{CR})$
  $\mu_F \leftarrow (1 - c) \cdot \mu_F + c \cdot lehmerMean(S_F)$
 end

Pseudocode: memetic JADE

Algorithm 5.1: Pseudocode of memetic JADE

Function `memeticJADE`($\mathbf{X}$, $funct$, $minErr$, $maxFE$):
    $dim, popsize \leftarrow size(\mathbf{X})$
    $p \leftarrow 0.3$
    $c \leftarrow 0.5$
    $pop, FE, F, CR \leftarrow JADE(\mathbf{X}, p, c, funct, minErr, maxFE - 2dim\ )$
    $bestIndex = argmin(FE)$
    $bestSol = pop[bestIndex]$
    $pop, FE = downhill\ simplex(funct, bestSol, minErr, 2dim)$
    return pop, FE, F, CR

Finite Elemente Results

Problem PDE	L2 Norm
0A	$2.967 \cdot 10^{-5}$
0B	$1.071 \cdot 10^{-5}$
1	$8.004 \cdot 10^{-7}$
2	$3.501 \cdot 10^{-8}$
3	$1.680 \cdot 10^{-9}$
4	$4.764 \cdot 10^{-7}$
5	$6.057 \cdot 10^{-6}$
6	$1.908 \cdot 10^{-7}$
7	$5.203 \cdot 10^{-5}$
8	$3.237 \cdot 10^{-7}$
9	$2.366 \cdot 10^{-7}$

Derivatives Kernel

Gauss Kernel

$$\frac{\partial u_{apx}(\mathbf{x})}{\partial x_j} = -2\sum_{i=0}^{N} \omega_i \gamma_i (x_j - c_{ij}) e^{-\gamma_i r_i^2} \tag{3.13}$$

$$\frac{\partial^2 u_{apx}(\mathbf{x})}{\partial x_j^2} = \sum_{i=0}^{N} \omega_i \gamma_i \left[4\gamma_i (x_j - c_{ij})^2 - 2\right] e^{-\gamma_i r_i^2} \tag{3.14}$$

$$\frac{\partial^2 u_{apx}(\mathbf{x})}{\partial x_j x_k} = 4\sum_{i=0}^{N} \omega_i \gamma_i^2 (x_j - c_{ij})(x_k - c_{ik}) e^{-\gamma_i r_i^2} \tag{3.15}$$

Gauss-Sine Kernel

$$\frac{\partial u_{apx}(\mathbf{x})}{\partial x_j} = \sum_{i=0}^{N} 2\omega_i (x_j - c_{ij}) e^{-\gamma_i r_i^2} (f_i cos(f_i r_i^2 - \varphi_i) - \gamma_i sin(f_i r_i^2 - \varphi_i)) \tag{3.19}$$

$$\frac{\partial^2 u_{apx}(\mathbf{x})}{\partial x_j^2} = \sum_{i=0}^{N} 2\omega_i e^{-\gamma_i r_i^2}$$
$$[-(2(f_i^2 - \gamma_i^2)(x_j - c_{ij})^2 + \gamma_i) sin(f_i r_i^2 - \varphi_i) - (4 f_i \gamma_i (x_j - c_{ij})^2 - f_i) cos(f_i r_i^2 - \varphi_i)] \tag{3.20}$$

$$\frac{\partial^2 u_{apx}(\mathbf{x})}{\partial x_j x_k} = \sum_{i=0}^{N} -4\omega_i (c_{ij} - x_j)(c_{ik} - x_k) e^{-\gamma_i r_i^2}$$
$$[(f_i^2 - \gamma_i^2) sin(f_i r_i^2 - \varphi_i) + 2 f_i \gamma_i cos(f_i r_i^2 - \varphi_i)] \tag{3.21}$$

Picture Library – revised