

RIDGE: An Autonomous Framework for Validation and Method Discovery in AI-Generated Option Pricing

Liexin Cheng^{1,2} Xue Cheng² Shuaiqiang Liu^{1,3}
Kees Oosterlee¹

¹Mathematical Institute, Utrecht University

²School of Mathematical Sciences, Peking University

³Delft Institute of Applied Mathematics, Delft University of Technology

October 2026

Agenda

- The RIDGE validation framework

Agenda

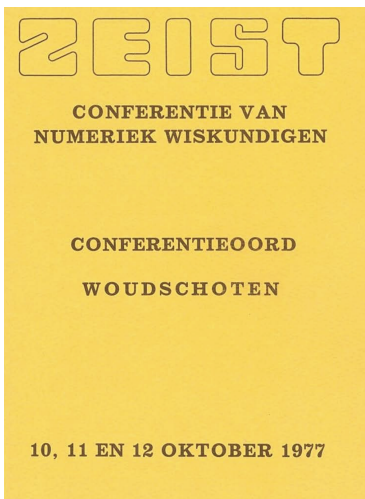
- The RIDGE validation framework
- Findings

Agenda

- The RIDGE validation framework
- Findings
- Method discovery

Agenda

- The RIDGE validation framework
- Findings
- Method discovery
- 50th Woudschoten



Organisers: Th. Dekker (UvA), W. Wetterling (Twente), M. Spijker (Leiden), E. Slagt (CWI):

ZEIST

CONFERENTIE VAN
NUMERIEK WISKUNDIGEN

CONFERENTIEOORD
WOUDSCHOTEN

10, 11 EN 12 OKTOBER 1977

Organisers: Th. Dekker (UvA), W. Wetterling (Twente), M. Spijker (Leiden), E. Slagt (CWI): “Numerical solution of nonlinear equations, en of integral equations ”

DEELNEMERS

1. P.M. Anselone	Universität Hamburg (tijdelijk)	Hamburg
2. C.T.H. Baker	University of Manchester	Manchester
3. F. Bakker	Rijksuniversiteit Leiden	Leiden
4. J.W. Boerstoeel	Nat. Lucht- en Ruimtevaart Lab.	Amsterdam
5. E. Bohl	Universität Münster	Münster
6. J. Bollen	Technische Hogeschool	Eindhoven
7. E.F.F. Botta	Rijksuniversiteit	Groningen
8. J.C.P. Bus	Mathematisch Centrum	Amsterdam
9. K. Dekker	Universiteit van Amsterdam	Amsterdam
10. Th.J. Dekker	Universiteit van Amsterdam	Amsterdam
11. L.M. Delves	University of Liverpool	Liverpool
12. P. Deuflhard	Technische Universität	München
13. A.W. den Exter		
Blokland	K.N.M.I.	de Bilt
14. P. Eijgenraam	Rijksuniversiteit	Leiden
15. H. Gerritsen	Technische Hogeschool Twente	Enschede
16. A.J. Geurts	Technische Hogeschool	Eindhoven
17. C. van Ginneken	Technische Hogeschool	Eindhoven
18. J.A. van de Griend	Rijksuniversiteit	Leiden
19. P.P.N. de Groen	Technische Hogeschool	Eindhoven
20. J. de Groot	Nat. Lab. Philips	Eindhoven
21. P.W. Hemker	Mathematisch Centrum	Amsterdam
22. C. den Heijer	Mathematisch Centrum	Amsterdam
23. W. Hoffmann	Universiteit van Amsterdam	Amsterdam
24. J.P. Hollenberg	Rijksuniversiteit (Rekencentrum)	Groningen
25. P.J. van der Houwen	Mathematisch Centrum	Amsterdam
26. J.K.M. Jansen	Technische Hogeschool	Eindhoven
27. J. Kok	Mathematisch Centrum	Amsterdam
28. C.G. van der Laan	Rijksuniversiteit (Rekencentrum)	Groningen
29. G.P. Leendertse	Energie Centrum Nederland	Petten
30. H.J. van Linde	Rijksuniversiteit (Rekencentrum)	Groningen
31. R.M.M. Mattheij	Kath. Universiteit	Nijmegen
32. J.J.H. Miller	Kath. Universiteit	Nijmegen
33. A.C.B. den Ouden	Energieonderzoek Centrum Nederland	Petten
34. J.J. Renes	Nat. Lucht- en Ruimtevaart Lab.	N.O.P.
35. H.J.J. te Riele	Mathematisch Centrum	Amsterdam
36. F. Robert	University of Grenoble	Grenoble
37. E.E. Schoone	Neratoom	Den Haag
38. J.L. Simons	Nat. Lucht- en Ruimtevaart Lab.	Amsterdam
39. E. Slagt	Mathematisch Centrum	Amsterdam
40. A. van de Sluis	Rijksuniversiteit	Utrecht
41. M.N. Spijker	Rijksuniversiteit	Leiden
42. C.R. Traas	Nat. Lucht- en Ruimtevaart Lab.	Amsterdam
43. M. van Veldhuizen	Vrije Universiteit	Amsterdam
44. A.E.P. Veldman	Nat. Lucht- en Ruimtevaart Lab.	Amsterdam
45. E.W. Veldkamp	Technische Hogeschool	Eindhoven
46. J.G. Verwer	Mathematisch Centrum	Amsterdam
47. A.I. van de Vooren	Rijksuniversiteit	Groningen
48. J.S. van Welij	Philips	Eindhoven
49. P. Wesseling	Technische Hogeschool	Delft
50. W. Wetterling	Technische Hogeschool Twente	Enschede
51. P.H.M. Wolkenfelt	Mathematisch Centrum	Amsterdam
52. L. Wuytack	Universitaire Instelling	Antwerpen
53. J.R. Zweerus	Rijksuniversiteit	Utrecht
54. J.J.M. Cuppen	Technische Hogeschool	Eindhoven
55. L.S. de Jong	Technische Hogeschool	Eindhoven

Scientific computing: a changing workflow

- For decades, the scientific-computing workflow has been:

mathematical model, numerical method, implementation → validation.

Scientific computing: a changing workflow

- For decades, the scientific-computing workflow has been:

mathematical model, numerical method, implementation $\rightarrow$ validation.

- Traditionally, deriving the model and numerical method required mathematical expertise, followed by substantial implementation effort.

Scientific computing: a changing workflow

- For decades, the scientific-computing workflow has been:

mathematical model, numerical method, implementation $\rightarrow$ validation.

- Traditionally, deriving the model and numerical method required mathematical expertise, followed by substantial implementation effort.
- Large language models are changing the workflow: from a model, they can directly generate substantial parts of a numerical implementation.

Scientific computing: a changing workflow

- For decades, the scientific-computing workflow has been:

mathematical model, numerical method, implementation $\rightarrow$ validation.

- Traditionally, deriving the model and numerical method required mathematical expertise, followed by substantial implementation effort.
- Large language models are changing the workflow: from a model, they can directly generate substantial parts of a numerical implementation.
- If numerical software is generated automatically, how should it be validated?

Scientific computing: a changing workflow

- For decades, the scientific-computing workflow has been:

mathematical model, numerical method, implementation → validation.

- Traditionally, deriving the model and numerical method required mathematical expertise, followed by substantial implementation effort.
 - Large language models are changing the workflow: from a model, they can directly generate substantial parts of a numerical implementation.
 - If numerical software is generated automatically, how should it be validated?
- ⇒ The bottleneck moves from code generation to scientific validation.

Validation as an operator problem

- Scientific software usually defines a **map between families of inputs and outputs**, not just a single numerical value.

Validation as an operator problem

- Scientific software usually defines a **map between families of inputs and outputs**, not just a single numerical value.
- Validation should therefore test the generated numerical operator
 - across parameter regimes;

Validation as an operator problem

- Scientific software usually defines a **map between families of inputs and outputs**, not just a single numerical value.
- Validation should therefore test the generated numerical operator
 - across parameter regimes;
 - across spatial or temporal resolutions;

Validation as an operator problem

- Scientific software usually defines a **map between families of inputs and outputs**, not just a single numerical value.
- Validation should therefore test the generated numerical operator
 - across parameter regimes;
 - across spatial or temporal resolutions;
 - across model configurations and stress cases.

Validation as an operator problem

- Scientific software usually defines a **map between families of inputs and outputs**, not just a single numerical value.
 - Validation should therefore test the generated numerical operator
 - across parameter regimes;
 - across spatial or temporal resolutions;
 - across model configurations and stress cases.
- ⇒ An operator viewpoint turns “does the code work?” into
“**does the generated map have the correct numerical behaviour?**”

A testbed: option pricing

- ⇒ Option pricing provides a convenient environment for studying generated numerical software.

A testbed: option pricing

- ⇒ Option pricing provides a convenient environment for studying generated numerical software.
- A European option has a terminal payoff $g(S_T)$, for example a call option $g(S_T) = (S_T - K)^+$.

A testbed: option pricing

- ⇒ Option pricing provides a convenient environment for studying generated numerical software.
- A European option has a terminal payoff $g(S_T)$, for example a call option $g(S_T) = (S_T - K)^+$.
 - Under the risk-neutral probability measure, the value is the discounted conditional expectation

$$V(t, S_t) = \mathbb{E}^{\mathbb{Q}} \left[e^{-r(T-t)} g(S_T) \mid S_t \right].$$

A testbed: option pricing

- ⇒ Option pricing provides a convenient environment for studying generated numerical software.
- A European option has a terminal payoff $g(S_T)$, for example a call option $g(S_T) = (S_T - K)^+$.
 - Under the risk-neutral probability measure, the value is the discounted conditional expectation

$$V(t, S_t) = \mathbb{E}^{\mathbb{Q}} \left[e^{-r(T-t)} g(S_T) \mid S_t \right].$$

- Numerically, this is useful because the solution can often be computed in several different ways:

Monte Carlo methods, PDE discretization, Fourier methods.

A testbed: option pricing

- ⇒ Option pricing provides a convenient environment for studying generated numerical software.
- A European option has a terminal payoff $g(S_T)$, for example a call option $g(S_T) = (S_T - K)^+$.
 - Under the risk-neutral probability measure, the value is the discounted conditional expectation

$$V(t, S_t) = \mathbb{E}^{\mathbb{Q}} \left[e^{-r(T-t)} g(S_T) \mid S_t \right].$$

- Numerically, this is useful because the solution can often be computed in several different ways:

Monte Carlo methods, PDE discretization, Fourier methods.

Feynman–Kac: expectation and PDE

- Consider the **Heston stochastic-volatility model**:

$$\begin{aligned}\frac{dS_t}{S_t} &= (r - q) dt + \sqrt{v_t} dW_1(t), \\ dv_t &= \kappa(\theta - v_t) dt + \sigma_v \sqrt{v_t} dW_2(t),\end{aligned}$$

with $d[W_1, W_2]_t = \rho dt$.

- Consider the **Heston stochastic-volatility model**:

$$\begin{aligned}\frac{dS_t}{S_t} &= (r - q) dt + \sqrt{v_t} dW_1(t), \\ dv_t &= \kappa(\theta - v_t) dt + \sigma_v \sqrt{v_t} dW_2(t),\end{aligned}$$

with $d[W_1, W_2]_t = \rho dt$.

- With $g(S_T) = V(t, S, v) = \mathbb{E}^{\mathbb{Q}} [e^{-r(T-t)} g(S_T) \mid S_t = S, v_t = v]$.

Feynman–Kac: expectation and PDE

- Consider the **Heston stochastic-volatility model**:

$$\begin{aligned}\frac{dS_t}{S_t} &= (r - q) dt + \sqrt{v_t} dW_1(t), \\ dv_t &= \kappa(\theta - v_t) dt + \sigma_v \sqrt{v_t} dW_2(t),\end{aligned}$$

with $d[W_1, W_2]_t = \rho dt$.

- With $g(S_T) = V(t, S, v) = \mathbb{E}^{\mathbb{Q}} [e^{-r(T-t)} g(S_T) \mid S_t = S, v_t = v]$.
- By the **Feynman–Kac theorem**, the same function solves

$$\begin{aligned}0 &= \partial_t V + (r - q)S \partial_S V + \kappa(\theta - v) \partial_v V \\ &\quad + \frac{1}{2} v S^2 \partial_{SS} V + \rho \sigma_v v S \partial_{Sv} V + \frac{1}{2} \sigma_v^2 v \partial_{vv} V - rV,\end{aligned}$$

with $V(T, S, v) = g(S)$.

RIDGE: an autonomous validation framework

RIDGE = **R**epository-driven **I**terative **D**iagnosis, **G**eneration and **E**valuation.

RIDGE: an autonomous validation framework

RIDGE = **R**epository-driven **I**terative **D**iagnosis, **G**eneration and **E**valuation.

- The objective of RIDGE is to validate generated numerical implementations in a systematic and reproducible manner.

RIDGE: an autonomous validation framework

RIDGE = **R**epository-driven **I**terative **D**iagnosis, **G**eneration and **E**valuation.

- The objective of RIDGE is to validate generated numerical implementations in a systematic and reproducible manner.
- Validation combines
 - deterministic numerical measurements;

RIDGE: an autonomous validation framework

RIDGE = **R**epository-driven **I**terative **D**iagnosis, **G**eneration and **E**valuation.

- The objective of RIDGE is to validate generated numerical implementations in a systematic and reproducible manner.
- Validation combines
 - deterministic numerical measurements;
 - diagnostic interpretation;

RIDGE: an autonomous validation framework

RIDGE = **R**epository-driven **I**terative **D**iagnosis, **G**eneration and **E**valuation.

- The objective of RIDGE is to validate generated numerical implementations in a systematic and reproducible manner.
- Validation combines
 - deterministic numerical measurements;
 - diagnostic interpretation;
 - iterative implementation refinement.

RIDGE: an autonomous validation framework

RIDGE = **R**epository-driven **I**terative **D**iagnosis, **G**eneration and **E**valuation.

- The objective of RIDGE is to validate generated numerical implementations in a systematic and reproducible manner.
- Validation combines
 - deterministic numerical measurements;
 - diagnostic interpretation;
 - iterative implementation refinement.
- The expertise is accumulated in an evolving **repository**, containing
 - benchmarks;
 - validation rules;
 - experience from previous validation iterations.

RIDGE: an autonomous validation framework

RIDGE = **R**epository-driven **I**terative **D**iagnosis, **G**eneration and **E**valuation.

- The objective of RIDGE is to validate generated numerical implementations in a systematic and reproducible manner.
 - Validation combines
 - deterministic numerical measurements;
 - diagnostic interpretation;
 - iterative implementation refinement.
 - The expertise is accumulated in an evolving **repository**, containing
 - benchmarks;
 - validation rules;
 - experience from previous validation iterations.
- ⇒ RIDGE is designed to guide **different coding agents** through the same validation logic.

A single RIDGE validation iteration

- A validation iteration starts from three ingredients.

A single RIDGE validation iteration

- A validation iteration starts from three ingredients.
 - A **blueprint**, specifying the mathematical model and numerical methodology.

A single RIDGE validation iteration

- A validation iteration starts from three ingredients.
 - A **blueprint**, specifying the mathematical model and numerical methodology.
 - A **generated implementation**, evaluated by numerical tests.

A single RIDGE validation iteration

- A validation iteration starts from three ingredients.
 - A **blueprint**, specifying the mathematical model and numerical methodology.
 - A **generated implementation**, evaluated by numerical tests.
 - A **repository**, containing previously validated knowledge.

A single RIDGE validation iteration

- A validation iteration starts from three ingredients.
 - A **blueprint**, specifying the mathematical model and numerical methodology.
 - A **generated implementation**, evaluated by numerical tests.
 - A **repository**, containing previously validated knowledge.
- The resulting evidence is interpreted diagnostically.

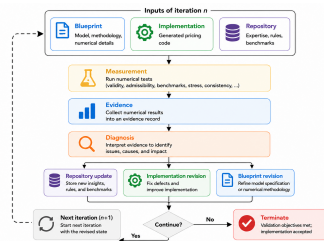
A single RIDGE validation iteration

- A validation iteration starts from three ingredients.
 - A **blueprint**, specifying the mathematical model and numerical methodology.
 - A **generated implementation**, evaluated by numerical tests.
 - A **repository**, containing previously validated knowledge.
- The resulting evidence is interpreted diagnostically.
- The diagnosis determines whether the implementation should be
 - repaired,
 - numerically refined,
 - or replaced by another methodology.

A single RIDGE validation iteration

- A validation iteration starts from three ingredients.
 - A **blueprint**, specifying the mathematical model and numerical methodology.
 - A **generated implementation**, evaluated by numerical tests.
 - A **repository**, containing previously validated knowledge.
- The resulting evidence is interpreted diagnostically.
- The diagnosis determines whether the implementation should be
 - repaired,
 - numerically refined,
 - or replaced by another methodology.

⇒ An **iterative learning process**.



Three levels of the numerical problem

- The mathematical model defines an exact operator

$$\mathcal{P} : \mathcal{M} \rightarrow \mathcal{Y}.$$

Three levels of the numerical problem

- The mathematical model defines an exact operator

$$\mathcal{P} : \mathcal{M} \rightarrow \mathcal{Y}.$$

- Discretization h gives a numerical target operator

$$\mathcal{P}_h : \mathcal{M}_h \rightarrow \mathcal{Y}_h.$$

Three levels of the numerical problem

- The mathematical model defines an exact operator

$$\mathcal{P} : \mathcal{M} \rightarrow \mathcal{Y}.$$

- Discretization h gives a numerical target operator

$$\mathcal{P}_h : \mathcal{M}_h \rightarrow \mathcal{Y}_h.$$

- The LLM-generated program realizes

$$\hat{\mathcal{P}}_h : \mathcal{M}_h \rightarrow \mathcal{Y}_h.$$

Three levels of the numerical problem

- The mathematical model defines an exact operator

$$\mathcal{P} : \mathcal{M} \rightarrow \mathcal{Y}.$$

- Discretization h gives a numerical target operator

$$\mathcal{P}_h : \mathcal{M}_h \rightarrow \mathcal{Y}_h.$$

- The LLM-generated program realizes

$$\hat{\mathcal{P}}_h : \mathcal{M}_h \rightarrow \mathcal{Y}_h.$$

- The central validation question is

Does $\hat{\mathcal{P}}_h$ faithfully realize $\mathcal{P}_h$ over the required domain?

Three levels of the numerical problem

- The mathematical model defines an exact operator

$$\mathcal{P} : \mathcal{M} \rightarrow \mathcal{Y}.$$

- Discretization h gives a numerical target operator

$$\mathcal{P}_h : \mathcal{M}_h \rightarrow \mathcal{Y}_h.$$

- The LLM-generated program realizes

$$\hat{\mathcal{P}}_h : \mathcal{M}_h \rightarrow \mathcal{Y}_h.$$

- The central validation question is

Does $\hat{\mathcal{P}}_h$ faithfully realize $\mathcal{P}_h$ over the required domain?

⇒ This separates the mathematical method from its generated implementation.

Implementation error versus discretization error

- The total error satisfies

$$\|\hat{\mathcal{P}}_h - \mathcal{P}\| \leq \underbrace{\|\hat{\mathcal{P}}_h - \mathcal{P}_h\|}_{\text{implementation error}} + \underbrace{\|\mathcal{P}_h - \mathcal{P}\|}_{\text{scheme error}}.$$

Implementation error versus discretization error

- The total error satisfies

$$\|\hat{\mathcal{P}}_h - \mathcal{P}\| \leq \underbrace{\|\hat{\mathcal{P}}_h - \mathcal{P}_h\|}_{\text{implementation error}} + \underbrace{\|\mathcal{P}_h - \mathcal{P}\|}_{\text{scheme error}}.$$

Implementation error

Did we realize the method?
Target of RIDGE

Scheme error

Is the numerical method accurate?
Classical numerical analysis

Implementation error versus discretization error

- The total error satisfies

$$\|\hat{\mathcal{P}}_h - \mathcal{P}\| \leq \underbrace{\|\hat{\mathcal{P}}_h - \mathcal{P}_h\|}_{\text{implementation error}} + \underbrace{\|\mathcal{P}_h - \mathcal{P}\|}_{\text{scheme error}}.$$

Implementation error

Did we realize the method?

Target of RIDGE

Scheme error

Is the numerical method accurate?

Classical numerical analysis

- RIDGE primarily attacks the first term.

Implementation error versus discretization error

- The total error satisfies

$$\|\hat{\mathcal{P}}_h - \mathcal{P}\| \leq \underbrace{\|\hat{\mathcal{P}}_h - \mathcal{P}_h\|}_{\text{implementation error}} + \underbrace{\|\mathcal{P}_h - \mathcal{P}\|}_{\text{scheme error}}.$$

Implementation error

Did we realize the method?

Target of RIDGE

Scheme error

Is the numerical method accurate?

Classical numerical analysis

- RIDGE primarily attacks the first term.
- Experiments will show that validation may discover that the **second term or the computational cost of the method** is the real bottleneck.

Implementation error versus discretization error

- The total error satisfies

$$\|\hat{\mathcal{P}}_h - \mathcal{P}\| \leq \underbrace{\|\hat{\mathcal{P}}_h - \mathcal{P}_h\|}_{\text{implementation error}} + \underbrace{\|\mathcal{P}_h - \mathcal{P}\|}_{\text{scheme error}}.$$

Implementation error	Scheme error
Did we realize the method? Target of RIDGE	Is the numerical method accurate? Classical numerical analysis

- RIDGE primarily attacks the first term.
 - Experiments will show that validation may discover that the **second term or the computational cost of the method** is the real bottleneck.
- ⇒ Successful implementation validation does not imply that the chosen numerical methodology is satisfactory.

Three validation blocks

- RIDGE organizes numerical evidence into three blocks.

Three validation blocks

- RIDGE organizes numerical evidence into three blocks.
- **Block I: Conformance**
Does the implementation execute correctly and realize the prescribed numerical methodology?

Three validation blocks

- RIDGE organizes numerical evidence into three blocks.
- **Block I: Conformance**
Does the implementation execute correctly and realize the prescribed numerical methodology?
- **Block II: Admissibility**
Does the generated solution satisfy mathematical structure, such as monotonicity, convexity and density conditions?

Three validation blocks

- RIDGE organizes numerical evidence into three blocks.
- **Block I: Conformance**
Does the implementation execute correctly and realize the prescribed numerical methodology?
- **Block II: Admissibility**
Does the generated solution satisfy mathematical structure, such as monotonicity, convexity and density conditions?
- **Block III: Benchmarking**
How closely does the generated solver agree with trusted reference solutions?

Three validation blocks

- RIDGE organizes numerical evidence into three blocks.
- **Block I: Conformance**
Does the implementation execute correctly and realize the prescribed numerical methodology?
- **Block II: Admissibility**
Does the generated solution satisfy mathematical structure, such as monotonicity, convexity and density conditions?
- **Block III: Benchmarking**
How closely does the generated solver agree with trusted reference solutions?
- Diagnosis then determines whether the issue is

[BUG], [METHOD-HYPER], [METHOD-ALGO], [METHOD-CHOICE].

Three validation blocks

- RIDGE organizes numerical evidence into three blocks.
- **Block I: Conformance**
Does the implementation execute correctly and realize the prescribed numerical methodology?
- **Block II: Admissibility**
Does the generated solution satisfy mathematical structure, such as monotonicity, convexity and density conditions?
- **Block III: Benchmarking**
How closely does the generated solver agree with trusted reference solutions?
- Diagnosis then determines whether the issue is

[BUG], [METHOD-HYPER], [METHOD-ALGO], [METHOD-CHOICE].

⇒ Measurement identifies symptoms; diagnosis determines the response.

RIDGE in action I: the Heston model

- Under the risk-neutral measure,

$$\frac{dS_t}{S_t} = (r - q) dt + \sqrt{v_t} dW_t,$$

$$dv_t = \kappa(\theta - v_t) dt + \sigma_v \sqrt{v_t} dZ_t, \quad d[W, Z]_t = \rho dt.$$

RIDGE in action I: the Heston model

- Under the risk-neutral measure,

$$\frac{dS_t}{S_t} = (r - q) dt + \sqrt{v_t} dW_t,$$

$$dv_t = \kappa(\theta - v_t) dt + \sigma_v \sqrt{v_t} dZ_t, \quad d[W, Z]_t = \rho dt.$$

- The characteristic function of $X_T = \log S_T$ is known analytically:

$$\varphi(u; T) = \mathbb{E} \left[e^{iuX_T} \right].$$

RIDGE in action I: the Heston model

- Under the risk-neutral measure,

$$\begin{aligned}\frac{dS_t}{S_t} &= (r - q) dt + \sqrt{v_t} dW_t, \\ dv_t &= \kappa(\theta - v_t) dt + \sigma_v \sqrt{v_t} dZ_t, \quad d[W, Z]_t = \rho dt.\end{aligned}$$

- The characteristic function of $X_T = \log S_T$ is known analytically:

$$\varphi(u; T) = \mathbb{E} \left[e^{iuX_T} \right].$$

- The generated blueprint therefore specifies

Heston characteristic function → Fourier COS valuation.

RIDGE in action I: the Heston model

- Under the risk-neutral measure,

$$\begin{aligned}\frac{dS_t}{S_t} &= (r - q) dt + \sqrt{v_t} dW_t, \\ dv_t &= \kappa(\theta - v_t) dt + \sigma_v \sqrt{v_t} dZ_t, \quad d[W, Z]_t = \rho dt.\end{aligned}$$

- The characteristic function of $X_T = \log S_T$ is known analytically:

$$\varphi(u; T) = \mathbb{E} \left[e^{iuX_T} \right].$$

- The generated blueprint therefore specifies

Heston characteristic function → Fourier COS valuation.

- This is an established and efficient methodology.
- ⇒ The first question is therefore purely a validation question: **is the generated realization numerically reliable?**

What does RIDGE test for Heston?

- Prices are computed on an (K, T) -grid, over many model configurations. The generated surface is checked for consequences of a valid option-pricing operator.

What does RIDGE test for Heston?

- Prices are computed on an (K, T) -grid, over many model configurations. The generated surface is checked for consequences of a valid option-pricing operator.
- Put-call parity:

$$V_c(t, K) - V_p(t, K) = S_t - Ke^{-r(T-t)}.$$

What does RIDGE test for Heston?

- Prices are computed on an (K, T) -grid, over many model configurations. The generated surface is checked for consequences of a valid option-pricing operator.
- Put-call parity:

$$V_c(t, K) - V_p(t, K) = S_t - Ke^{-r(T-t)}.$$

- Monotonicity:

$$V_c(t, K_{j+1}) \leq V_c(t, K_j).$$

What does RIDGE test for Heston?

- Prices are computed on an (K, T) -grid, over many model configurations. The generated surface is checked for consequences of a valid option-pricing operator.

- Put-call parity:

$$V_c(t, K) - V_p(t, K) = S_t - Ke^{-r(T-t)}.$$

- Monotonicity:

$$V_c(t, K_{j+1}) \leq V_c(t, K_j).$$

- Convexity:

$$V_c(t, K_{j-1}) - 2V_c(t, K_j) + V_c(t, K_{j+1}) \geq 0.$$

What does RIDGE test for Heston?

- Prices are computed on an (K, T) -grid, over many model configurations. The generated surface is checked for consequences of a valid option-pricing operator.

- Put-call parity:

$$V_c(t, K) - V_p(t, K) = S_t - Ke^{-r(T-t)}.$$

- Monotonicity:

$$V_c(t, K_{j+1}) \leq V_c(t, K_j).$$

- Convexity:

$$V_c(t, K_{j-1}) - 2V_c(t, K_j) + V_c(t, K_{j+1}) \geq 0.$$

- Density consistency:

$$f_Q(T, K) = e^{r(T-t)} \frac{\partial^2 V_c(t, K)}{\partial K^2} \geq 0, \quad \int_0^\infty f_Q(T, K) dK = 1.$$

What does RIDGE test for Heston?

- Prices are computed on an (K, T) -grid, over many model configurations. The generated surface is checked for consequences of a valid option-pricing operator.

- Put-call parity:

$$V_c(t, K) - V_p(t, K) = S_t - Ke^{-r(T-t)}.$$

- Monotonicity:

$$V_c(t, K_{j+1}) \leq V_c(t, K_j).$$

- Convexity:

$$V_c(t, K_{j-1}) - 2V_c(t, K_j) + V_c(t, K_{j+1}) \geq 0.$$

- Density consistency:

$$f_Q(T, K) = e^{r(T-t)} \frac{\partial^2 V_c(t, K)}{\partial K^2} \geq 0, \quad \int_0^\infty f_Q(T, K) dK = 1.$$

⇒ These tests probe analytical consistency, local structure and global quality of the price surface.

Heston: the first generated implementation fails

- The first implementation contains the correct Heston characteristic function and the correct Fourier pricing formula.

Heston: the first generated implementation fails

- The first implementation contains the correct Heston characteristic function and the correct Fourier pricing formula.
- But the numerical choices are essentially taken as fixed recipes from the literature:
 - a basic, fixed integration range;

Heston: the first generated implementation fails

- The first implementation contains the correct Heston characteristic function and the correct Fourier pricing formula.
- But the numerical choices are essentially taken as fixed recipes from the literature:
 - a basic, fixed integration range;
 - a fixed number of cosine terms;

Heston: the first generated implementation fails

- The first implementation contains the correct Heston characteristic function and the correct Fourier pricing formula.
- But the numerical choices are essentially taken as fixed recipes from the literature:
 - a basic, fixed integration range;
 - a fixed number of cosine terms;
 - the same numerical settings across all parameter regimes.

Heston: the first generated implementation fails

- The first implementation contains the correct Heston characteristic function and the correct Fourier pricing formula.
- But the numerical choices are essentially taken as fixed recipes from the literature:
 - a basic, fixed integration range;
 - a fixed number of cosine terms;
 - the same numerical settings across all parameter regimes.
- RIDGE finds that this is not robust:
 - the computational domain may be too small in the tails;

Heston: the first generated implementation fails

- The first implementation contains the correct Heston characteristic function and the correct Fourier pricing formula.
- But the numerical choices are essentially taken as fixed recipes from the literature:
 - a basic, fixed integration range;
 - a fixed number of cosine terms;
 - the same numerical settings across all parameter regimes.
- RIDGE finds that this is not robust:
 - the computational domain may be too small in the tails;
 - relevant evaluation points may lie too close to its boundary;

Heston: the first generated implementation fails

- The first implementation contains the correct Heston characteristic function and the correct Fourier pricing formula.
- But the numerical choices are essentially taken as fixed recipes from the literature:
 - a basic, fixed integration range;
 - a fixed number of cosine terms;
 - the same numerical settings across all parameter regimes.
- RIDGE finds that this is not robust:
 - the computational domain may be too small in the tails;
 - relevant evaluation points may lie too close to its boundary;
 - a fixed spectral resolution may be insufficient;

Heston: the first generated implementation fails

- The first implementation contains the correct Heston characteristic function and the correct Fourier pricing formula.
- But the numerical choices are essentially taken as fixed recipes from the literature:
 - a basic, fixed integration range;
 - a fixed number of cosine terms;
 - the same numerical settings across all parameter regimes.
- RIDGE finds that this is not robust:
 - the computational domain may be too small in the tails;
 - relevant evaluation points may lie too close to its boundary;
 - a fixed spectral resolution may be insufficient;
 - very small option values may suffer from numerical cancellation;

Heston: the first generated implementation fails

- The first implementation contains the correct Heston characteristic function and the correct Fourier pricing formula.
- But the numerical choices are essentially taken as fixed recipes from the literature:
 - a basic, fixed integration range;
 - a fixed number of cosine terms;
 - the same numerical settings across all parameter regimes.
- RIDGE finds that this is not robust:
 - the computational domain may be too small in the tails;
 - relevant evaluation points may lie too close to its boundary;
 - a fixed spectral resolution may be insufficient;
 - very small option values may suffer from numerical cancellation;
 - stressed parameters expose errors invisible in standard test cases.

Heston: the first generated implementation fails

- The first implementation contains the correct Heston characteristic function and the correct Fourier pricing formula.
- But the numerical choices are essentially taken as fixed recipes from the literature:
 - a basic, fixed integration range;
 - a fixed number of cosine terms;
 - the same numerical settings across all parameter regimes.
- RIDGE finds that this is not robust:
 - the computational domain may be too small in the tails;
 - relevant evaluation points may lie too close to its boundary;
 - a fixed spectral resolution may be insufficient;
 - very small option values may suffer from numerical cancellation;
 - stressed parameters expose errors invisible in standard test cases.
- The initial validation score is only 31/100.

Heston: the first generated implementation fails

- The first implementation contains the correct Heston characteristic function and the correct Fourier pricing formula.
 - But the numerical choices are essentially taken **as fixed recipes from the literature**:
 - a basic, fixed integration range;
 - a fixed number of cosine terms;
 - the same numerical settings across all parameter regimes.
 - RIDGE finds that this is not robust:
 - the computational domain may be too small in the tails;
 - relevant evaluation points may lie too close to its boundary;
 - a fixed spectral resolution may be insufficient;
 - very small option values may suffer from numerical cancellation;
 - stressed parameters expose errors invisible in standard test cases.
 - The initial validation score is only **31/100**.
- ⇒ **A correct formula does not imply a reliable numerical implementation.**

Heston: validation generates numerical adaptivity

- Successive validation iterations replace the fixed numerical choices by **adaptive ones**:
 - the computational domain is adapted to the model distribution;

Heston: validation generates numerical adaptivity

- Successive validation iterations replace the fixed numerical choices by **adaptive ones**:
 - the computational domain is adapted to the model distribution;
 - all requested evaluation points are explicitly covered;

Heston: validation generates numerical adaptivity

- Successive validation iterations replace the fixed numerical choices by **adaptive ones**:
 - the computational domain is adapted to the model distribution;
 - all requested evaluation points are explicitly covered;
 - the spectral resolution is increased when needed;

Heston: validation generates numerical adaptivity

- Successive validation iterations replace the fixed numerical choices by **adaptive ones**:
 - the computational domain is adapted to the model distribution;
 - all requested evaluation points are explicitly covered;
 - the spectral resolution is increased when needed;
 - the domain is enlarged in stressed parameter regimes;

Heston: validation generates numerical adaptivity

- Successive validation iterations replace the fixed numerical choices by **adaptive ones**:
 - the computational domain is adapted to the model distribution;
 - all requested evaluation points are explicitly covered;
 - the spectral resolution is increased when needed;
 - the domain is enlarged in stressed parameter regimes;
 - numerically stable representations are used for very small values.

Heston: validation generates numerical adaptivity

- Successive validation iterations replace the fixed numerical choices by **adaptive ones**:
 - the computational domain is adapted to the model distribution;
 - all requested evaluation points are explicitly covered;
 - the spectral resolution is increased when needed;
 - the domain is enlarged in stressed parameter regimes;
 - numerically stable representations are used for very small values.
- The principle is simple: **Numerical parameters should adapt to the problem, rather than be fixed once and for all.**

Heston: validation generates numerical adaptivity

- Successive validation iterations replace the fixed numerical choices by **adaptive ones**:
 - the computational domain is adapted to the model distribution;
 - all requested evaluation points are explicitly covered;
 - the spectral resolution is increased when needed;
 - the domain is enlarged in stressed parameter regimes;
 - numerically stable representations are used for very small values.
- The principle is simple: **Numerical parameters should adapt to the problem, rather than be fixed once and for all.**
- After ten validation iterations: 31/100 → 97/100.

Heston: validation generates numerical adaptivity

- Successive validation iterations replace the fixed numerical choices by **adaptive ones**:
 - the computational domain is adapted to the model distribution;
 - all requested evaluation points are explicitly covered;
 - the spectral resolution is increased when needed;
 - the domain is enlarged in stressed parameter regimes;
 - numerically stable representations are used for very small values.
- The principle is simple: **Numerical parameters should adapt to the problem, rather than be fixed once and for all.**
- After ten validation iterations: 31/100 → **97/100.**
- One complete maturity slice is priced in approximately 0.8 ms.

The repository begins to contain numerical expertise

- After the Heston experiment, the repository contains reusable numerical rules:
 - how to choose a sufficiently large computational domain;

The repository begins to contain numerical expertise

- After the Heston experiment, the repository contains reusable numerical rules:
 - how to choose a sufficiently large computational domain;
 - how to guarantee coverage of the requested evaluation region;

The repository begins to contain numerical expertise

- After the Heston experiment, the repository contains reusable numerical rules:
 - how to choose a sufficiently large computational domain;
 - how to guarantee coverage of the requested evaluation region;
 - how to adapt spectral resolution;

The repository begins to contain numerical expertise

- After the Heston experiment, the repository contains reusable numerical rules:
 - how to choose a sufficiently large computational domain;
 - how to guarantee coverage of the requested evaluation region;
 - how to adapt spectral resolution;
 - how to recognize parameter regimes requiring extra resolution;

The repository begins to contain numerical expertise

- After the Heston experiment, the repository contains reusable numerical rules:
 - how to choose a sufficiently large computational domain;
 - how to guarantee coverage of the requested evaluation region;
 - how to adapt spectral resolution;
 - how to recognize parameter regimes requiring extra resolution;
 - how to avoid unstable representations of small quantities.

The repository begins to contain numerical expertise

- After the Heston experiment, the repository contains reusable numerical rules:
 - how to choose a sufficiently large computational domain;
 - how to guarantee coverage of the requested evaluation region;
 - how to adapt spectral resolution;
 - how to recognize parameter regimes requiring extra resolution;
 - how to avoid unstable representations of small quantities.
- The next models therefore do not start from zero.

Model	Initial score	Final score
Heston model	31/100	97/100
Bates model	58/100	96/100
rough Heston model	32/100	99/100

The repository begins to contain numerical expertise

- After the Heston experiment, the repository contains reusable numerical rules:
 - how to choose a sufficiently large computational domain;
 - how to guarantee coverage of the requested evaluation region;
 - how to adapt spectral resolution;
 - how to recognize parameter regimes requiring extra resolution;
 - how to avoid unstable representations of small quantities.
- The next models therefore do not start from zero.

Model	Initial score	Final score
Heston model	31/100	97/100
Bates model	58/100	96/100
rough Heston model	32/100	99/100

⇒ Validated knowledge transfers from one generated solver to another.

RIDGE in action II: Heston–Hull–White

- The next experiment adds stochastic interest rates:

$$\frac{dS_t}{S_t} = r_t dt + \sqrt{v_t} dW_1(t),$$

$$dv_t = \kappa(\eta - v_t) dt + \sigma_1 \sqrt{v_t} dW_2(t),$$

$$dr_t = a(b - r_t) dt + \sigma_2 dW_3(t).$$

RIDGE in action II: Heston–Hull–White

- The next experiment adds stochastic interest rates:

$$\frac{dS_t}{S_t} = r_t dt + \sqrt{v_t} dW_1(t),$$

$$dv_t = \kappa(\eta - v_t) dt + \sigma_1 \sqrt{v_t} dW_2(t),$$

$$dr_t = a(b - r_t) dt + \sigma_2 dW_3(t).$$

- The three driving Brownian motions may be correlated.

RIDGE in action II: Heston–Hull–White

- The next experiment adds stochastic interest rates:

$$\frac{dS_t}{S_t} = r_t dt + \sqrt{v_t} dW_1(t),$$

$$dv_t = \kappa(\eta - v_t) dt + \sigma_1 \sqrt{v_t} dW_2(t),$$

$$dr_t = a(b - r_t) dt + \sigma_2 dW_3(t).$$

- The three driving Brownian motions may be correlated.
- This destroys the simple exponential–affine characteristic-function representation available in Heston.

RIDGE in action II: Heston–Hull–White

- The next experiment adds stochastic interest rates:

$$\frac{dS_t}{S_t} = r_t dt + \sqrt{v_t} dW_1(t),$$

$$dv_t = \kappa(\eta - v_t) dt + \sigma_1 \sqrt{v_t} dW_2(t),$$

$$dr_t = a(b - r_t) dt + \sigma_2 dW_3(t).$$

- The three driving Brownian motions may be correlated.
- This destroys the simple exponential–affine characteristic-function representation available in Heston.
- The first generated blueprint therefore selects **Euler Monte Carlo**.

RIDGE in action II: Heston–Hull–White

- The next experiment adds stochastic interest rates:

$$\frac{dS_t}{S_t} = r_t dt + \sqrt{v_t} dW_1(t),$$

$$dv_t = \kappa(\eta - v_t) dt + \sigma_1 \sqrt{v_t} dW_2(t),$$

$$dr_t = a(b - r_t) dt + \sigma_2 dW_3(t).$$

- The three driving Brownian motions may be correlated.
 - This destroys the simple exponential–affine characteristic-function representation available in Heston.
 - The first generated blueprint therefore selects **Euler Monte Carlo**.
- ⇒ This case tests whether RIDGE can distinguish between a poor implementation and a poor numerical methodology.

HHW: first RIDGE repairs the Monte Carlo implementation

- The first iterations reveal genuine implementation problems.

HHW: first RIDGE repairs the Monte Carlo implementation

- The first iterations reveal genuine implementation problems.
- **Short-rate initialization:** the initial rate is not made consistent with the market rate. A known limiting case exposes the error.

HHW: first RIDGE repairs the Monte Carlo implementation

- The first iterations reveal genuine implementation problems.
- **Short-rate initialization:** the initial rate is not made consistent with the market rate. A known limiting case exposes the error.
- **Variance discretization:** partial truncation of the CIR variance process behaves poorly in stressed regimes and is replaced by full truncation.

HHW: first RIDGE repairs the Monte Carlo implementation

- The first iterations reveal genuine implementation problems.
- **Short-rate initialization:** the initial rate is not made consistent with the market rate. A known limiting case exposes the error.
- **Variance discretization:** partial truncation of the CIR variance process behaves poorly in stressed regimes and is replaced by full truncation.
- The validation score improves

$$38/120 \rightarrow 53/120.$$

HHW: first RIDGE repairs the Monte Carlo implementation

- The first iterations reveal genuine implementation problems.
- **Short-rate initialization:** the initial rate is not made consistent with the market rate. A known limiting case exposes the error.
- **Variance discretization:** partial truncation of the CIR variance process behaves poorly in stressed regimes and is replaced by full truncation.
- The validation score improves

$$38/120 \rightarrow 53/120.$$

- At this stage RIDGE behaves exactly as in Heston:

measure $\rightarrow$ diagnose $\rightarrow$ repair.

HHW: first RIDGE repairs the Monte Carlo implementation

- The first iterations reveal genuine implementation problems.
- **Short-rate initialization:** the initial rate is not made consistent with the market rate. A known limiting case exposes the error.
- **Variance discretization:** partial truncation of the CIR variance process behaves poorly in stressed regimes and is replaced by full truncation.
- The validation score improves

$$38/120 \rightarrow 53/120.$$

- At this stage RIDGE behaves exactly as in Heston:

measure $\rightarrow$ diagnose $\rightarrow$ repair.

- After these defects are removed, the residual error pattern changes.

HHW: when the method itself becomes the problem

- The repaired Monte Carlo implementation is internally consistent.

HHW: when the method itself becomes the problem

- The repaired Monte Carlo implementation is internally consistent.
- Nevertheless:
 - further code revisions give little improvement;

HHW: when the method itself becomes the problem

- The repaired Monte Carlo implementation is internally consistent.
- Nevertheless:
 - further code revisions give little improvement;
 - statistical noise decreases only slowly;

HHW: when the method itself becomes the problem

- The repaired Monte Carlo implementation is internally consistent.
- Nevertheless:
 - further code revisions give little improvement;
 - statistical noise decreases only slowly;
 - higher accuracy becomes computationally expensive.

HHW: when the method itself becomes the problem

- The repaired Monte Carlo implementation is internally consistent.
- Nevertheless:
 - further code revisions give little improvement;
 - statistical noise decreases only slowly;
 - higher accuracy becomes computationally expensive.
- The fundamental Monte Carlo rate is

$$\text{sampling error} = \mathcal{O}(N_{\text{MC}}^{-1/2}).$$

HHW: when the method itself becomes the problem

- The repaired Monte Carlo implementation is internally consistent.
- Nevertheless:
 - further code revisions give little improvement;
 - statistical noise decreases only slowly;
 - higher accuracy becomes computationally expensive.
- The fundamental Monte Carlo rate is

$$\text{sampling error} = \mathcal{O}(N_{\text{MC}}^{-1/2}).$$

- Reducing the statistical error by a factor 100 requires roughly 10^4 times as many paths.

HHW: when the method itself becomes the problem

- The repaired Monte Carlo implementation is internally consistent.
- Nevertheless:
 - further code revisions give little improvement;
 - statistical noise decreases only slowly;
 - higher accuracy becomes computationally expensive.
- The fundamental Monte Carlo rate is

$$\text{sampling error} = \mathcal{O}(N_{\text{MC}}^{-1/2}).$$

- Reducing the statistical error by a factor 100 requires roughly 10^4 times as many paths.
- This is no longer evidence for [BUG] or [METHOD-HYPER].

[METHOD-CHOICE]

HHW: when the method itself becomes the problem

- The repaired Monte Carlo implementation is internally consistent.
- Nevertheless:
 - further code revisions give little improvement;
 - statistical noise decreases only slowly;
 - higher accuracy becomes computationally expensive.
- The fundamental Monte Carlo rate is

$$\text{sampling error} = \mathcal{O}(N_{\text{MC}}^{-1/2}).$$

- Reducing the statistical error by a factor 100 requires roughly 10^4 times as many paths.
- This is no longer evidence for [BUG] or [METHOD-HYPER].

[METHOD-CHOICE]

⇒ RIDGE concludes that the numerical methodology, rather than its implementation, is the bottleneck.

HHW: from Monte Carlo to a one-dimensional PDE

- RIDGE therefore searches for an alternative mathematical representation.

HHW: from Monte Carlo to a one-dimensional PDE

- RIDGE therefore searches for an alternative mathematical representation.
- When $\rho_{23} = 0$, conditioning on the variance process v_t makes the remaining stock and interest-rate contributions Gaussian.

HHW: from Monte Carlo to a one-dimensional PDE

- RIDGE therefore searches for an alternative mathematical representation.
- When $\rho_{23} = 0$, conditioning on the variance process v_t makes the remaining stock and interest-rate contributions Gaussian.
- These contributions can be integrated out analytically, so the remaining conditional expectation depends only on v_t .

HHW: from Monte Carlo to a one-dimensional PDE

- RIDGE therefore searches for an alternative mathematical representation.
- When $\rho_{23} = 0$, conditioning on the variance process v_t makes the remaining stock and interest-rate contributions Gaussian.
- These contributions can be integrated out analytically, so the remaining conditional expectation depends only on v_t .
- Feynman–Kac then gives a one-dimensional PDE for a characteristic function ψ :

$$\partial_t \psi + \kappa(\eta - v) \partial_v \psi + \frac{1}{2} \sigma_1^2 v \partial_{vv} \psi + c_{\text{HHW}}(u, t, v) \psi = 0.$$

HHW: from Monte Carlo to a one-dimensional PDE

- RIDGE therefore searches for an alternative mathematical representation.
- When $\rho_{23} = 0$, conditioning on the variance process v_t makes the remaining stock and interest-rate contributions Gaussian.
- These contributions can be integrated out analytically, so the remaining conditional expectation depends only on v_t .
- Feynman–Kac then gives a one-dimensional PDE for a characteristic function ψ :

$$\partial_t \psi + \kappa(\eta - v) \partial_v \psi + \frac{1}{2} \sigma_1^2 v \partial_{vv} \psi + c_{\text{HHW}}(u, t, v) \psi = 0.$$

- Solving this PDE for the required Fourier frequencies provides the characteristic function for the final pricing step.

HHW: from Monte Carlo to a one-dimensional PDE

- RIDGE therefore searches for an alternative mathematical representation.
- When $\rho_{23} = 0$, conditioning on the variance process v_t makes the remaining stock and interest-rate contributions Gaussian.
- These contributions can be integrated out analytically, so the remaining conditional expectation depends only on v_t .
- Feynman–Kac then gives a one-dimensional PDE for a characteristic function ψ :

$$\partial_t \psi + \kappa(\eta - v) \partial_v \psi + \frac{1}{2} \sigma_1^2 v \partial_{vv} \psi + c_{\text{HHW}}(u, t, v) \psi = 0.$$

- Solving this PDE for the required Fourier frequencies provides the characteristic function for the final pricing step.

Monte Carlo $\rightarrow$ 1D PDE $\rightarrow$ Fourier pricing.

HHW: validation and final outcome

- Changing methodology creates a **new numerical target operator**.

HHW: validation and final outcome

- Changing methodology creates a **new numerical target operator**.
- The PDE–Fourier implementation can reuse numerical knowledge from Heston, but the inherited rules are tested again.

HHW: validation and final outcome

- Changing methodology creates a **new numerical target operator**.
- The PDE–Fourier implementation can reuse numerical knowledge from Heston, but the inherited rules are tested again.
- Validation now adapts the PDE domain and grid, the Fourier resolution, and the final numerical evaluation.

HHW: validation and final outcome

- Changing methodology creates a **new numerical target operator**.
- The PDE–Fourier implementation can reuse numerical knowledge from Heston, but the inherited rules are tested again.
- Validation now adapts the PDE domain and grid, the Fourier resolution, and the final numerical evaluation.
- The final benchmark discrepancy is

$$\max_T |\Delta IV_{ATM}| = 5.07 \times 10^{-5},$$

while parity, monotonicity and convexity violations are reduced to essentially machine precision.

HHW: validation and final outcome

- Changing methodology creates a **new numerical target operator**.
- The PDE–Fourier implementation can reuse numerical knowledge from Heston, but the inherited rules are tested again.
- Validation now adapts the PDE domain and grid, the Fourier resolution, and the final numerical evaluation.
- The final benchmark discrepancy is

$$\max_T |\Delta IV_{\text{ATM}}| = 5.07 \times 10^{-5},$$

while parity, monotonicity and convexity violations are reduced to essentially machine precision.

- The diagnostic score improves from

$$38/120 \rightarrow 118/120.$$

HHW: validation and final outcome

- Changing methodology creates a **new numerical target operator**.
- The PDE–Fourier implementation can reuse numerical knowledge from Heston, but the inherited rules are tested again.
- Validation now adapts the PDE domain and grid, the Fourier resolution, and the final numerical evaluation.
- The final benchmark discrepancy is

$$\max_T |\Delta IV_{ATM}| = 5.07 \times 10^{-5},$$

while parity, monotonicity and convexity violations are reduced to essentially machine precision.

- The diagnostic score improves from

$$38/120 \rightarrow 118/120.$$

⇒ **RIDGE can repair an implementation and, when necessary, change the numerical methodology itself.**

Method discovery as a change of target operator

- Initially RIDGE tries to validate

$$\hat{\mathcal{P}}_h^{\text{MC}} \approx \mathcal{P}_h^{\text{MC}}.$$

Method discovery as a change of target operator

- Initially RIDGE tries to validate

$$\hat{\mathcal{P}}_h^{\text{MC}} \approx \mathcal{P}_h^{\text{MC}}.$$

- Implementation defects are removed, but the remaining limitation belongs to the methodology itself.

Method discovery as a change of target operator

- Initially RIDGE tries to validate

$$\hat{\mathcal{P}}_h^{\text{MC}} \approx \mathcal{P}_h^{\text{MC}}.$$

- Implementation defects are removed, but the remaining limitation belongs to the methodology itself.
- The numerical target is therefore changed:

$$\mathcal{P}_h^{\text{MC}} \longrightarrow \mathcal{P}_h^{\text{PDE/Fourier}}.$$

Method discovery as a change of target operator

- Initially RIDGE tries to validate

$$\hat{\mathcal{P}}_h^{\text{MC}} \approx \mathcal{P}_h^{\text{MC}}.$$

- Implementation defects are removed, but the remaining limitation belongs to the methodology itself.
- The numerical target is therefore changed:

$$\mathcal{P}_h^{\text{MC}} \longrightarrow \mathcal{P}_h^{\text{PDE/Fourier}}.$$

- RIDGE then faces a new validation problem:

$$\hat{\mathcal{P}}_h^{\text{PDE/Fourier}} \approx \mathcal{P}_h^{\text{PDE/Fourier}}.$$

Method discovery as a change of target operator

- Initially RIDGE tries to validate

$$\hat{\mathcal{P}}_h^{\text{MC}} \approx \mathcal{P}_h^{\text{MC}}.$$

- Implementation defects are removed, but the remaining limitation belongs to the methodology itself.
- The numerical target is therefore changed:

$$\mathcal{P}_h^{\text{MC}} \longrightarrow \mathcal{P}_h^{\text{PDE/Fourier}}.$$

- RIDGE then faces a new validation problem:

$$\hat{\mathcal{P}}_h^{\text{PDE/Fourier}} \approx \mathcal{P}_h^{\text{PDE/Fourier}}.$$

⇒ Method discovery means changing the target numerical operator, followed by a new validation iteration.

A message for numerical analysis

- Generated scientific software changes the role of numerical validation.

A message for numerical analysis

- Generated scientific software changes the role of numerical validation.
- Many scientific-computing problems have the structure

model $\rightarrow$ discretization $\rightarrow$ generated implementation $\rightarrow$ validation.

A message for numerical analysis

- Generated scientific software changes the role of numerical validation.
- Many scientific-computing problems have the structure

model $\rightarrow$ discretization $\rightarrow$ generated implementation $\rightarrow$ validation.

- The validation tests are application dependent: they may involve conservation, positivity, monotonicity, symmetry, invariants, or convergence under mesh refinement.

A message for numerical analysis

- Generated scientific software changes the role of numerical validation.
- Many scientific-computing problems have the structure

model $\rightarrow$ discretization $\rightarrow$ generated implementation $\rightarrow$ validation.

- The validation tests are application dependent: they may involve conservation, positivity, monotonicity, symmetry, invariants, or convergence under mesh refinement.
- It's important to exploit the **mathematical structure of the problem**.

A message for numerical analysis

- Generated scientific software changes the role of numerical validation.
- Many scientific-computing problems have the structure

model $\rightarrow$ discretization $\rightarrow$ generated implementation $\rightarrow$ validation.

- The validation tests are application dependent: they may involve conservation, positivity, monotonicity, symmetry, invariants, or convergence under mesh refinement.
- It's important to exploit the **mathematical structure of the problem**.
- In our experiments, validation drives

implementation repair $\rightarrow$ numerical adaptivity $\rightarrow$ method selection.

A message for numerical analysis

- Generated scientific software changes the role of numerical validation.
- Many scientific-computing problems have the structure

model $\rightarrow$ discretization $\rightarrow$ generated implementation $\rightarrow$ validation.

- The validation tests are application dependent: they may involve conservation, positivity, monotonicity, symmetry, invariants, or convergence under mesh refinement.
- It's important to exploit the **mathematical structure of the problem**.
- In our experiments, validation drives

implementation repair $\rightarrow$ numerical adaptivity $\rightarrow$ method selection.

- Different coding agents may arrive at different numerical solutions, while the validation framework provides a common scientific standard.

A message for numerical analysis

- Generated scientific software changes the role of numerical validation.
- Many scientific-computing problems have the structure

model $\rightarrow$ discretization $\rightarrow$ generated implementation $\rightarrow$ validation.

- The validation tests are application dependent: they may involve conservation, positivity, monotonicity, symmetry, invariants, or convergence under mesh refinement.
- It's important to exploit the **mathematical structure of the problem**.
- In our experiments, validation drives

implementation repair $\rightarrow$ numerical adaptivity $\rightarrow$ method selection.

- Different coding agents may arrive at different numerical solutions, while the validation framework provides a common scientific standard.
- $\Rightarrow$ **Validation may become an active component of numerical-method development and discovery.**

50 editions of Woudschoten: the first generation

- A strong tradition in **Numerical Mathematics**: Axelsson, Dekker, Hemker, Spijker, van der Sluis, Verwer, Veltkamp, van de Vooren, van der Vorst, van der Houwen, Wesseling, ...

50 editions of Woudschoten: the first generation

- A strong tradition in **Numerical Mathematics**: Axelsson, Dekker, Hemker, Spijker, van der Sluis, Verwer, Veltkamp, van de Vooren, van der Vorst, van der Houwen, Wesseling, . . .
- Numerical analysis and numerical linear algebra, with **PDEs, discretization and computational fluid dynamics** as major themes.

50 editions of Woudschoten: the first generation

- A strong tradition in **Numerical Mathematics**: Axelsson, Dekker, Hemker, Spijker, van der Sluis, Verwer, Veltkamp, van de Vooren, van der Vorst, van der Houwen, Wesseling, ...
- Numerical analysis and numerical linear algebra, with **PDEs, discretization and computational fluid dynamics** as major themes.
- Strong interaction between **universities, CWI and industry**.

50 editions of Woudschoten: the first generation

- A strong tradition in **Numerical Mathematics**: Axelsson, Dekker, Hemker, Spijker, van der Sluis, Verwer, Velthkamp, van de Vooren, van der Vorst, van der Houwen, Wesseling, . . .
 - Numerical analysis and numerical linear algebra, with **PDEs, discretization and computational fluid dynamics** as major themes.
 - Strong interaction between **universities, CWI and industry**.
- ⇒ **The foundations of the Woudschoten community.**
- A next generation carried Woudschoten forward towards Scientific Computing: Bisseling, Hundsdorfer, Koren, Mattheij, Roose, Schilders, Vandewalle, van der Vegt, Veldman, Vuik, . . .

50 editions of Woudschoten: the first generation

- A strong tradition in **Numerical Mathematics**: Axelsson, Dekker, Hemker, Spijker, van der Sluis, Verwer, Velthkamp, van de Vooren, van der Vorst, van der Houwen, Wesseling, . . .
 - Numerical analysis and numerical linear algebra, with **PDEs, discretization and computational fluid dynamics** as major themes.
 - Strong interaction between **universities, CWI and industry**.
- ⇒ **The foundations of the Woudschoten community.**
- A next generation carried Woudschoten forward towards Scientific Computing: Bisseling, Hundsdorfer, Koren, Mattheij, Roose, Schilders, Vandewalle, van der Vegt, Veldman, Vuik, . . .
 - More diverse applications, stochastic problems, integral equations and high-dimensional computation.

50 editions of Woudschoten: the first generation

- A strong tradition in **Numerical Mathematics**: Axelsson, Dekker, Hemker, Spijker, van der Sluis, Verwer, Velthkamp, van de Vooren, van der Vorst, van der Houwen, Wesseling, . . .
 - Numerical analysis and numerical linear algebra, with **PDEs, discretization and computational fluid dynamics** as major themes.
 - Strong interaction between **universities, CWI and industry**.
- ⇒ **The foundations of the Woudschoten community.**
- A next generation carried Woudschoten forward towards Scientific Computing: Bisseling, Hundsdorfer, Koren, Mattheij, Roose, Schilders, Vandewalle, van der Vegt, Veldman, Vuik, . . .
 - More diverse applications, stochastic problems, integral equations and high-dimensional computation.
 - **1996: Flanders formally joins Woudschoten.**



Woudschoten today

- A **new generation** is carrying Woudschoten forward:
 - v. Brummelen, Crommelin, Vandaele, Dolean, Dubinkina, Frank, Gerritsma, van Gijzen, in 't Hout, Lord, Meerbergen, Oosterlee, Pop, Sanderse, Schlottbom, Stevenson, Vermolen, Veroy, Verstappen,
 - ...

Woudschoten today

- A **new generation** is carrying Woudschoten forward:
 - v. Brummelen, Crommelin, Vandaele, Dolean, Dubinkina, Frank, Gerritsma, van Gijzen, in 't Hout, Lord, Meerbergen, Oosterlee, Pop, Sanderse, Schlottbom, Stevenson, Vermolen, Veroy, Verstappen,
...
- The field can no longer be captured under one or two headings.

Woudschoten today

- A **new generation** is carrying Woudschoten forward:
 - v. Brummelen, Crommelin, Vandaele, Dolean, Dubinkina, Frank, Gerritsma, van Gijzen, in 't Hout, Lord, Meerbergen, Oosterlee, Pop, Sanderse, Schlottbom, Stevenson, Vermolen, Veroy, Verstappen,
...
- The field can no longer be captured under one or two headings.
- **AI, high-dimensional problems and quantum computing** alongside beautiful classical numerical problems.

Woudschoten today

- A **new generation** is carrying Woudschoten forward:
 - v. Brummelen, Crommelin, Vandaele, Dolean, Dubinkina, Frank, Gerritsma, van Gijzen, in 't Hout, Lord, Meerbergen, Oosterlee, Pop, Sanderse, Schlottbom, Stevenson, Vermolen, Veroy, Verstappen,
...
- The field can no longer be captured under one or two headings.
- **AI, high-dimensional problems and quantum computing** alongside beautiful classical numerical problems.
- Universities, CWI and **industry** continue to meet at Woudschoten.

Woudschoten today

- A **new generation** is carrying Woudschoten forward:
 - v. Brummelen, Crommelin, Vandaele, Dolean, Dubinkina, Frank, Gerritsma, van Gijzen, in 't Hout, Lord, Meerbergen, Oosterlee, Pop, Sanderse, Schlottbom, Stevenson, Vermolen, Veroy, Verstappen,
...
 - The field can no longer be captured under one or two headings.
 - **AI, high-dimensional problems and quantum computing** alongside beautiful classical numerical problems.
 - Universities, CWI and **industry** continue to meet at Woudschoten.
- ⇒ **A changing field, but still one Scientific Computing community.**

Towards the 75th edition

- Keeping a community like Woudschoten strong is perhaps less self-evident than it once was:
 - fewer PhD students, a broader field, and increasingly expensive conference venues.

Towards the 75th edition

- Keeping a community like Woudschoten strong is perhaps less self-evident than it once was:
 - fewer PhD students, a broader field, and increasingly expensive conference venues.
- Yet Woudschoten continues to offer something very special:
 - beautiful lectures and lessons from experts;

Towards the 75th edition

- Keeping a community like Woudschoten strong is perhaps less self-evident than it once was:
 - fewer PhD students, a broader field, and increasingly expensive conference venues.
- Yet Woudschoten continues to offer something very special:
 - beautiful lectures and lessons from experts;
 - a valuable network across universities, CWI and industry;

Towards the 75th edition

- Keeping a community like Woudschoten strong is perhaps less self-evident than it once was:
 - fewer PhD students, a broader field, and increasingly expensive conference venues.
- Yet Woudschoten continues to offer something very special:
 - beautiful lectures and lessons from experts;
 - a valuable network across universities, CWI and industry;
 - young researchers meeting and learning from experienced colleagues.

Towards the 75th edition

- Keeping a community like Woudschoten strong is perhaps less self-evident than it once was:
 - fewer PhD students, a broader field, and increasingly expensive conference venues.
- Yet Woudschoten continues to offer something very special:
 - beautiful lectures and lessons from experts;
 - a valuable network across universities, CWI and industry;
 - young researchers meeting and learning from experienced colleagues.
- With gratitude to CWI for its continued support, and to the Woudschoten secretaries who helped sustain our community: Evert Slagt, Jan Verwer, Willem Hundsorfer, Ben Sommeijer, Jan Kok, Walter Lioen, Margreet Nool, Martine Anholt.

Towards the 75th edition

- Keeping a community like Woudschoten strong is perhaps less self-evident than it once was:
 - fewer PhD students, a broader field, and increasingly expensive conference venues.
- Yet Woudschoten continues to offer something very special:
 - beautiful lectures and lessons from experts;
 - a valuable network across universities, CWI and industry;
 - young researchers meeting and learning from experienced colleagues.
- With gratitude to CWI for its continued support, and to the Woudschoten secretaries who helped sustain our community: Evert Slagt, Jan Verwer, Willem Hundsdofer, Ben Sommeijer, Jan Kok, Walter Lioen, Margreet Nool, Martine Anholt.

On to the 75th edition!