

um
2026

AI-Driven Insights in modeFRONTIER

Prototyping the AI Copilot assistant and the Pareto navigator

Enrico Rigoni
Head of Optimization and Modeling



Agenda

Introduction

Guiding questions

AI Copilot assistant

Delivering factually grounded insights

Pareto navigator

Translating human intent into optimal decisions



Introduction



Introduction

Integrating natural language interaction within modeFRONTIER is a key step toward democratizing complex optimization technology.

Guiding questions:

- How might we guide a user from a problem description to a running plan?
- How might we help user in configuring optimization algorithms?
- How might we help users in getting insights from simulation/optimization results?

To deliver actionable AI-driven insights, our team is developing two distinct prototypes: the **AI Copilot assistant** and the **Pareto navigator**.



AI Copilot assistant



AI Copilot assistant

Aim:

Develop an AI Copilot assistant that goes beyond traditional documentation lookup, enabling more intelligent and efficient interactions.

Users can ask the AI Copilot assistant any question about their plan and algorithm configurations to obtain factually grounded answers. For general workflow inquiries the AI Copilot assistant consults the standard user manual, but for deeper questions it can consult more specialized documents (such as the PDF files of our technical reports).

Deployment architectures: cloud vs. local

Cloud LLMs

Pros: Maximum reasoning capability, zero local infrastructure overhead.

Cons: Requires internet connectivity and strict data governance.

Local SLMs & Air-gapped

Pros: Complete data sovereignty and offline capability.

Cons: Trade-offs include high local hardware requirements (GPUs) and reduced out-of-the-box reasoning power.

The main engineering challenges

Data Security & Privacy

- Preventing proprietary IP from becoming training data.

Hallucinations

- The tendency of LLMs to confidently invent plausible but incorrect facts.

Context Windows & Costs

- Parsing massive project files (like JSON plans) wastes tokens and degrades performance.

UX & Latency

- Preventing slow AI network calls from freezing desktop application UI threads.

Our solution architecture

- Cloud-based API with **BYOK (Bring Your Own Key)** for security.
- **RAG (Retrieval-Augmented Generation)** to ground answers in truth.

Our engineering assistant implementation

Objective

- Integrate Google's Gemini (2.0 Flash) directly into the Planner environment in modeFRONTIER

Core Technology

- Retrieval-Augmented Generation (RAG) pipeline

Key Focus Areas

- Native UI performance
- Enterprise security
- Zero *AI hallucinations*

Four levels of depth

Level 1

- Reading the Plan configuration (JSON file) for basic questions about problem and algorithm configuration

Level 2

- Reading user manuals (HTML files) for general workflow inquiries

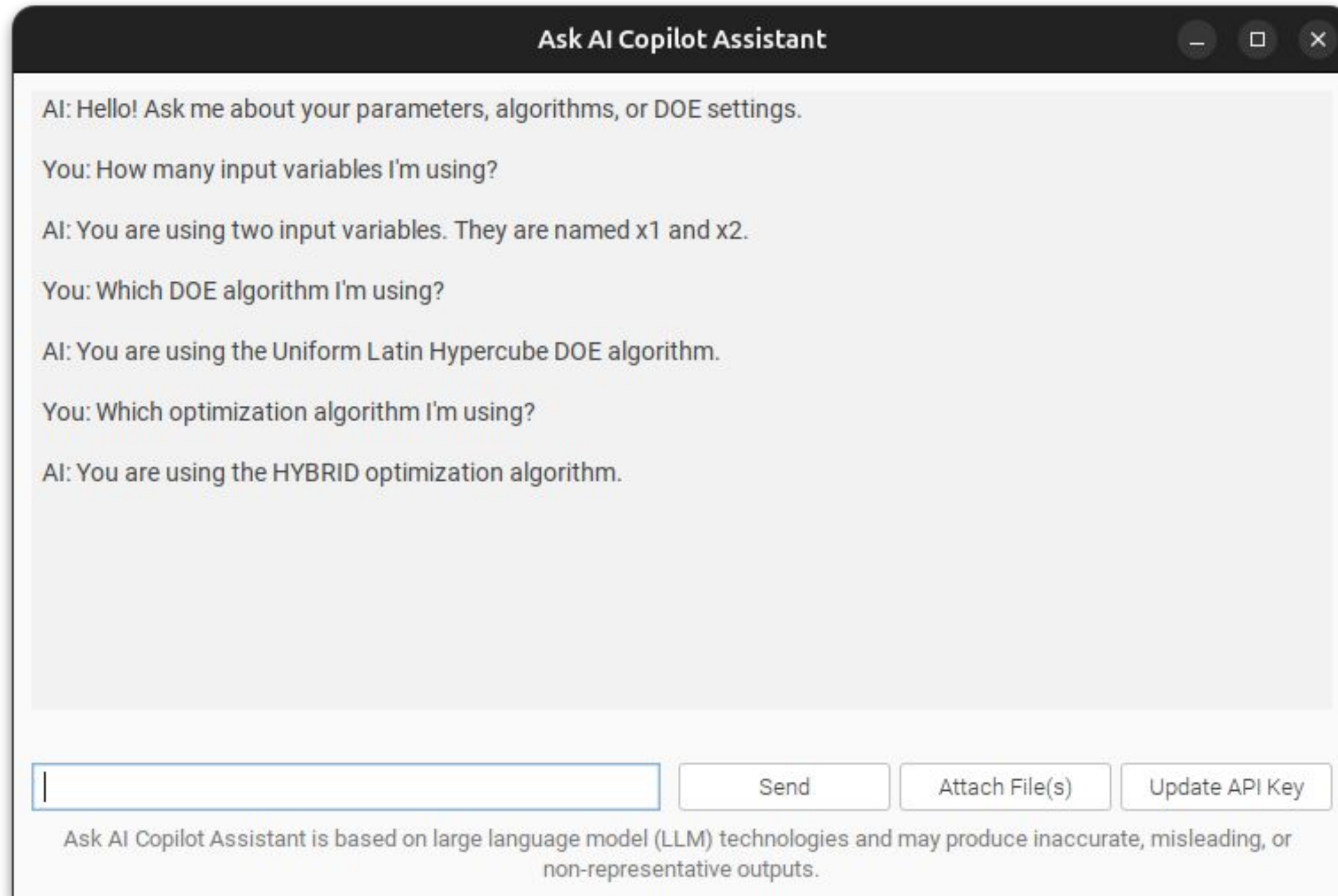
Level 3

- Reading technical reports (PDF files) for deep technical insights

Level 4

- Reading additional documentation provided by the user for precise answers

Conversation example



The screenshot shows a chat window titled "Ask AI Copilot Assistant". The conversation is as follows:

AI: Hello! Ask me about your parameters, algorithms, or DOE settings.

You: How many input variables I'm using?

AI: You are using two input variables. They are named x1 and x2.

You: Which DOE algorithm I'm using?

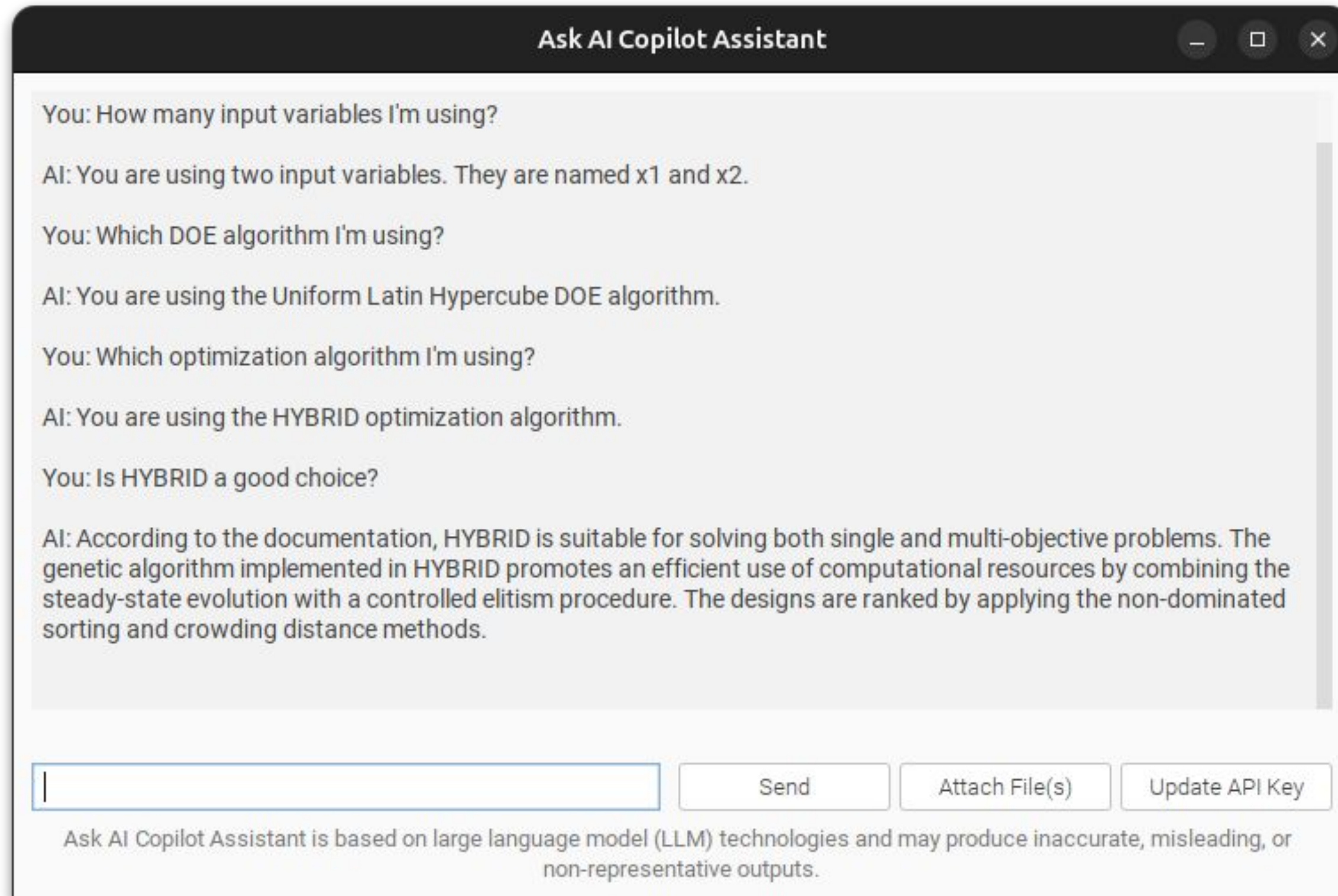
AI: You are using the Uniform Latin Hypercube DOE algorithm.

You: Which optimization algorithm I'm using?

AI: You are using the HYBRID optimization algorithm.

At the bottom of the window, there is an input field with a cursor, and three buttons: "Send", "Attach File(s)", and "Update API Key". Below the input field, a disclaimer reads: "Ask AI Copilot Assistant is based on large language model (LLM) technologies and may produce inaccurate, misleading, or non-representative outputs."

Conversation example



The screenshot shows a chat window titled "Ask AI Copilot Assistant". The conversation is as follows:

You: How many input variables I'm using?

AI: You are using two input variables. They are named x1 and x2.

You: Which DOE algorithm I'm using?

AI: You are using the Uniform Latin Hypercube DOE algorithm.

You: Which optimization algorithm I'm using?

AI: You are using the HYBRID optimization algorithm.

You: Is HYBRID a good choice?

AI: According to the documentation, HYBRID is suitable for solving both single and multi-objective problems. The genetic algorithm implemented in HYBRID promotes an efficient use of computational resources by combining the steady-state evolution with a controlled elitism procedure. The designs are ranked by applying the non-dominated sorting and crowding distance methods.

At the bottom of the window, there is an input field with a cursor, and three buttons: "Send", "Attach File(s)", and "Update API Key". Below the input field, a disclaimer reads: "Ask AI Copilot Assistant is based on large language model (LLM) technologies and may produce inaccurate, misleading, or non-representative outputs."

Conversation example

Ask AI Copilot Assistant

You: Could you show me some benchmark results of HYBRID algorithm?

AI: According to the technical report "Hybrid - Benchmark Tests", the HYBRID algorithm was tested on six different problems: ZDT4, UP2, SymPart, OSY, CTP2, and Lennard-Jones 10-atom cluster.

- ZDT4: MFGA obtained the best results in the first stages of the optimization run, while NSGA-2 prevails at the end. All Hybrid versions show performances in between (or very close to) the two standard GA. The forward differences implementation is the less effective.
- UP2: The algorithms behave as in the previous case: Hybrid is always very close to the best GA, metamodels give good and cheap gradient approximations and finite differences are forced to waste design evaluations.
- SymPart: The Sym-Part problem highlights the benefits of Hybrid, which outperforms classical GA. The high number of input variables penalizes the use of finite differences, while RBF approximation seems to work particularly well on this problem.
- OSY: Hybrid is competitive also on this problem. The performances of the five considered implementations are quite similar, but after half of the optimization run the SVD metamodel seems to provide very accurate search directions and at the end this version of Hybrid outperforms all other algorithms.
- CTP2: The RBF Hybrid is the clear winner of this benchmark test.
- Lennard-Jones 10-atom cluster: The performance of the Finite Difference Hybrid implementation are the best ones for this benchmark. This reflects the smoothness of the problem.

The technical report concludes that Hybrid is a robust algorithm which can solve accurately a wide range of problems.

Ask AI Copilot Assistant is based on large language model (LLM) technologies and may produce inaccurate, misleading, or non-representative outputs.

Conversation example

Ask AI Copilot Assistant

You: It is possible to reverse the direction of entropy growth?

AI: I do not have documentation on that topic. Please use the Attach File(s) button to provide the relevant PDF or document.

SYSTEM NOTE: Successfully attached 1 file(s).

- Entropy Has No Direction A Mirror-State Paradox Against Universal Monotonic Entropy Increase and a First-Principles...2026.pdf

You can now ask your question, and the AI will analyze these specific documents.

You: It is possible to reverse the direction of entropy growth?

AI: According to the available information, including the attached documentation, the strict (deterministic) form of the Second Law says that for an isolated system, entropy does not decrease along every trajectory, i.e. $S(t + \delta t) \geq S(t)$ for all t and all microstates (or $dS/dt \geq 0$ wherever defined). The statistical form states that for an appropriate ensemble or limit, entropy increase is "overwhelmingly probable" or the ensemble-average entropy change is nonnegative. The document argues that such universal monotonicity claims are logically incompatible with time-reversal symmetry. The consistent replacement is that entropy has no direction; it is described by a probability distribution. Moreover, constraints and boundary conditions reshape this distribution.

Send Attach File(s) Update API Key

Ask AI Copilot Assistant is based on large language model (LLM) technologies and may produce inaccurate, misleading, or non-representative outputs.

Pareto navigator



Pareto navigator




The Pareto navigator focuses on the final decision-making stage. It allows users to express their design preferences in natural language, translating human intent into the precise selection of optimal solutions within the Design Space.

Background

- Decisions often involve conflicting objectives
- Selecting the best designs can be time-consuming
- With 2–3 objectives, visual selection is manageable
- Beyond 3 objectives, decision-making becomes challenging

Benefit matrix

	≤ 3 objectives	> 3 objectives
Few designs	Limited	Moderate
Many designs	Moderate	High

-  **Limited:** Direct table-based selection is often enough
-  **Moderate:** Visual inspection or direct comparison is still manageable
-  **High:** Strong support for exploring trade-offs and recommending designs

Problem setting

Given:

- A dataset of evaluated candidate designs
- A preference description provided by the Decision Maker (DM)

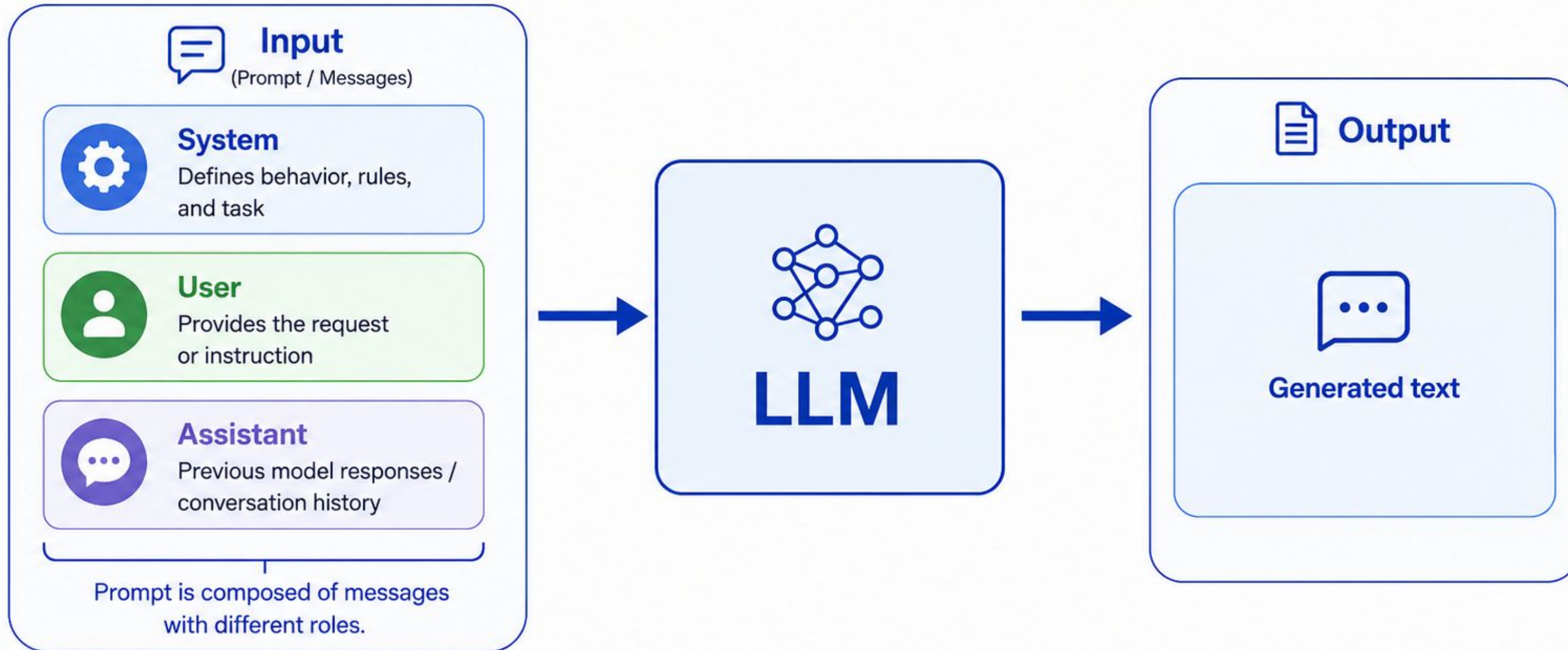
The system must return:

- A recommended design or set of designs
- A rationale explaining why those designs were selected
- **In a conversational setting:** The system may iteratively request feedback until satisfactory solutions are identified

Additional requirement:

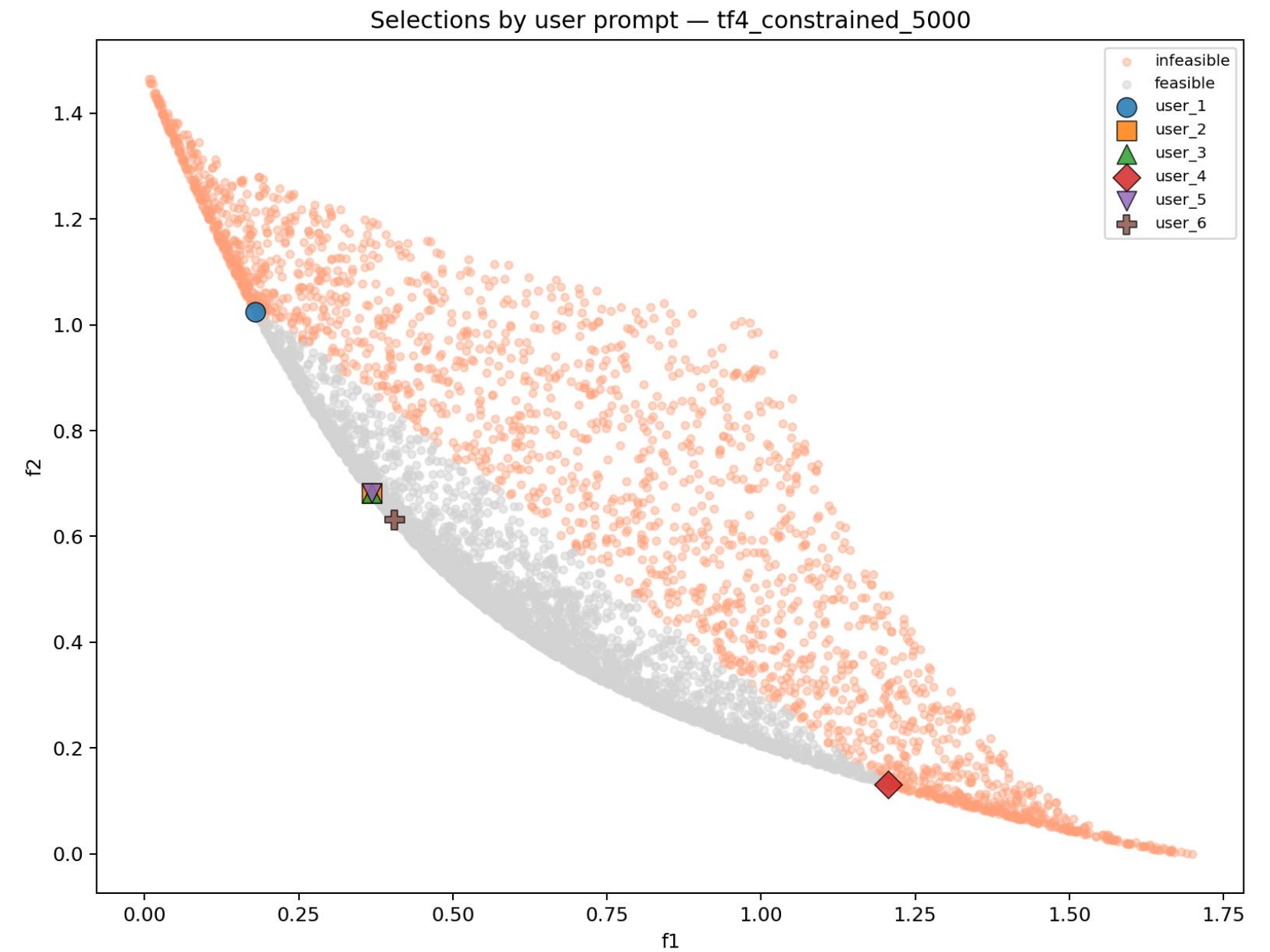
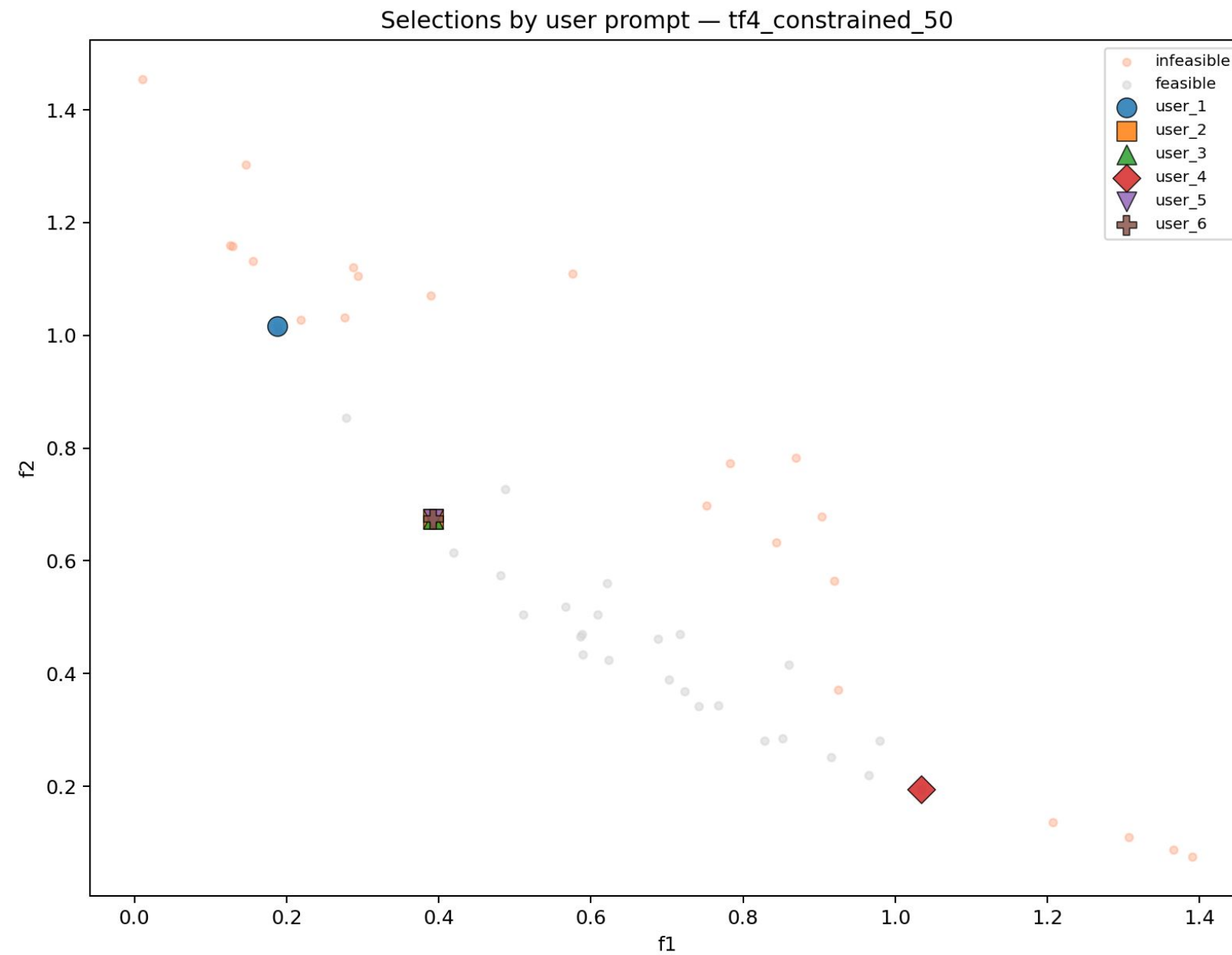
- The system must handle unclear preferences, input errors, and unrelated requests

Process flow



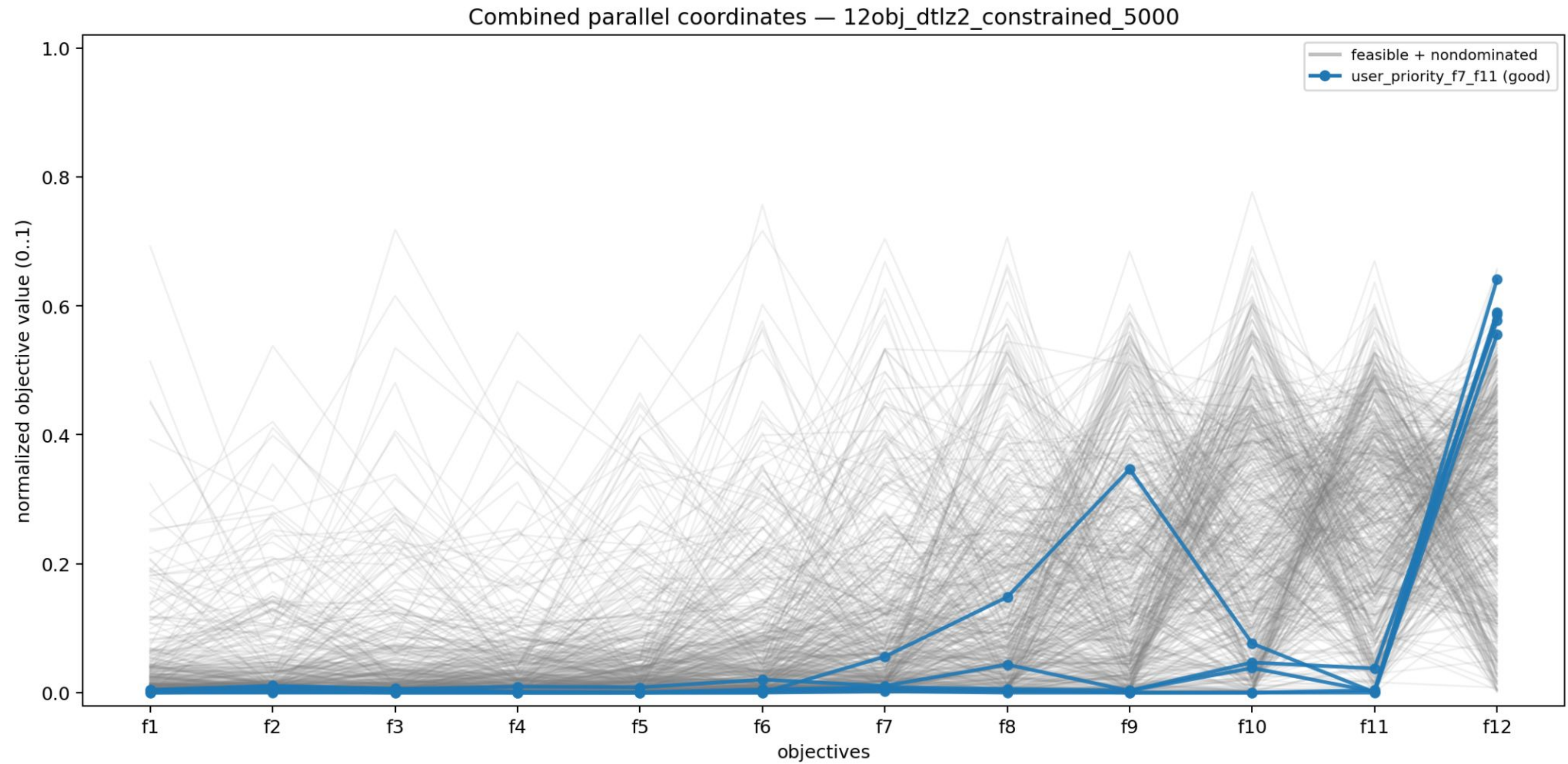
i The model receives the sequence of messages as context and generates the next assistant response.

Results - 2d example



- user_1: Prioritize f1 over f2
- user_2: Prioritize f1 over f2; knee-type tradeoff
- user_3: Knee-type tradeoff; avoid extremes
- user_4: Prioritize f2 over f1
- user_5: Prioritize f2 over f1; knee-type tradeoff
- user_6: Minimize f2 with f1 close to 0.4

Results - 12d example



Conversation example

Conversational Pareto Navigator — session 2026-05-20T15-47-02__12obj_dtlz2_constrained_5000

Dataset: datasets/12obj_dtlz2_constrained_5000.csv (rows=5000)

Objectives: ["f1", "f2", "f3", "f4", "f5", "f6", "f7", "f8", "f9", "f10", "f11", "f12"]

Type /help for commands. Press Ctrl+D to exit.

To get multiple designs, say so in your preference (e.g. 'show 3 tradeoffs'). Default is 1.

USER · TURN 0

Hi, I'd like to prioritize f4, f7 and f11. Can you recommend to me 4 options?

PARETO NAVIGATOR

--- Turn 0 ---

Selected 4 designs:

[1]

SELECTED DESIGN 2903

id	f4	f7	f11
2903	1.98e-09	5.55e-05	0.00539

Rationale: Corner for f4 with the lowest f4=1.98e-09 and very low f7=5.55e-05 and f11=0.00539, ideal if minimizing f4 is paramount while still keeping f7 and f11 extremely small.

[2]

SELECTED DESIGN 3602

id	f4	f7	f11
3602	4.78e-06	1.25e-05	0.955

Rationale: Corner for f7 with the lowest f7=1.25e-05 and very low f4=4.78e-06, but f11=0.955 is high; good if f7 and f4 dominate and you can tolerate large f11.

[3]

Conversation example

USER · TURN 1

Nice, from that selection what solution provides the best tradeoff over the objectives?

PARETO NAVIGATOR

--- Turn 1 ---

SELECTED DESIGN 1217

id	f4	f7	f11
1217	7.22e-05	2.05e-04	9.49e-04

Rationale: Among the four, id=1217 best balances all three objectives with $f4=7.22e-05$, $f7=2.05e-04$, and $f11=9.49e-04$, staying very low on each instead of optimizing just one while letting another become large.

(pool=652, shown=64, n_select=1, latency=4.90s)

Conversation example

USER - TURN 2

Thank you. I'd like you to put now f1 and f12 as priorities instead. Give me 2 options

PARETO NAVIGATOR

--- Turn 2 ---

Selected 2 designs:

[1]

SELECTED DESIGN 2903

id	f1	f12
2903	1.14e-09	1.44724

Rationale: Corner for f1 with the minimum $f1=1.14e-09$ but very high $f12=1.44724$; best if minimizing f1 is paramount and you can tolerate a large f12.

[2]

SELECTED DESIGN 3509

id	f1	f12
3509	0.00774	0.00364

Rationale: Corner for f12 with the minimum $f12=0.00364$ and low $f1=0.00774$; strong opposite tradeoff to 2903, prioritizing f12 while keeping f1 small.

(pool=652, shown=63, n_select=2, latency=3.52s)

Conversation example

USER · TURN 3

maybe an option that has a good tradeoff between both objectives

PARETO NAVIGATOR

--- Turn 3 ---

SELECTED DESIGN 3643

id	f1	f12
3643	0.000604	0.00384

Rationale: Id=3643 is the knee solution with $f1=0.000604$ and $f12=0.00384$, giving a very balanced and jointly low compromise between both objectives compared with points that strongly favor only one of them.

(pool=652, shown=62, n_select=1, latency=4.96s)

Conversation example

USER · TURN 4

Thank you! now I'd like to discuss the recipe of the tiramisu

PARETO NAVIGATOR

--- Turn 4 ---

I'm a Pareto Navigator assistant. I can only help you select tradeoff designs from this dataset.

(reason: The request is about a dessert recipe rather than selecting tradeoff designs from this optimization dataset.)

Try a preference like "minimize f1" or "show me 3 balanced designs".

Conversation example

USER · TURN 4

No problem! then give me solutions as priority f50

PARETO NAVIGATOR

--- Turn 4 ---

I'm a Pareto Navigator assistant. I can only help you select tradeoff designs from this dataset.

(reason: The requested objective f50 does not exist in this dataset, which only has objectives f1 through f12.)

Try a preference like "minimize f1" or "show me 3 balanced designs".

um
2026

Thank you

[esteco.com](https://www.esteco.com)

