

OPTIMIZATION GUIDE

How to Evaluate and Benchmark Mathematical Optimization Solvers.

A structured, three-phase plan for a confident, well-informed solver selection.

Purpose of this guide.

Every optimization model is unique, and a thoughtful evaluation process is essential for making a confident, well-informed solver selection. No matter the complexity of your project or the stage you are in, this guide is designed to help you set up a clear, structured plan to evaluate a mathematical optimization solver, including the Gurobi Optimizer.

Why use this guide.

Using a clearly documented evaluation process helps build trust across technical and business stakeholders, making solver selection easier to communicate and justify internally. This guide is designed to make solver evaluation easier and more transparent. It will help you:



Define a clear process for documenting solver comparisons and performance benchmarking.



Minimize effort and risk by focusing on targeted, representative tests.

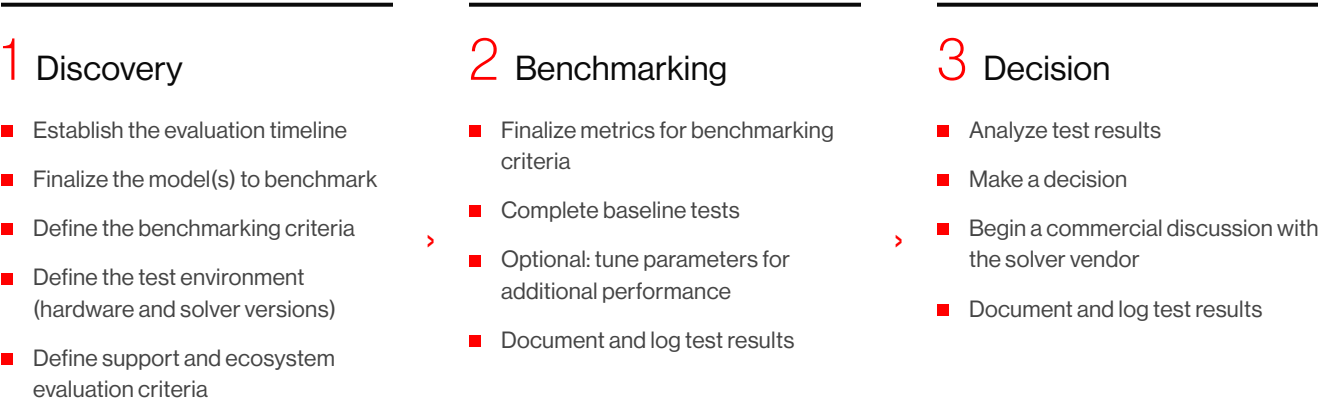


Evaluate support resources and ecosystem maturity.

At Gurobi, we use the term *evaluation* to describe the entire engagement process for trying new solvers, while *benchmarking* refers specifically to solver performance testing with defined metrics. An evaluation typically includes some form of benchmarking in addition to discussions around deployment architecture and licensing. Although this guide highlights the Gurobi evaluation process, we encourage you to use this same structure when evaluating other solvers, as it supports consistent comparisons and internal alignment.

Overview of the evaluation and benchmarking process.

To help you plan your solver evaluation, we have outlined three key phases, each with specific activities, stakeholders, and decision points. This structure makes it easier to coordinate across teams, align expectations, and track progress from discovery to decision.



Participants and collaboration.

Phase	Participants	Check-ins with Gurobi
1. Discovery	Technical Users & Stakeholders	Kickoff call to discuss evaluation goals, support needs, and your deployment scenario
2. Benchmarking	Technical Users	Technical sync to discuss performance and results
3. Decision	Technical Users & Stakeholders	Finalize analysis of benchmarking results and determine path forward



FIRST PHASE

Discovery.

During the discovery phase, you may wish to discuss some or all of the activities below with your Gurobi account team to support planning, evaluation design, and alignment on next steps.

If you have not yet selected a concrete use case for optimization, or if your team is not yet set up to develop a mathematical model, your Gurobi Technical Account Manager, Sales Account Manager, or our Certified Partner Network can help you get started.

Key activities.

a. Establish the evaluation timeline

Align with your Gurobi technical team on expected check-in frequency and preferred communication cadence throughout the evaluation. This can be as minimal or involved as you need.

Suggested reading: [How does a Gurobi Trial and Evaluation work?](#)

b. Finalize the model(s) to benchmark

Running tests on a model that is still under active development may lead to misleading results and unnecessary rework. Similarly, running tests on a model formulation that has been customized to favor another solver may not provide a fair basis for comparison.

Therefore, it is important to:

- Ensure model formulations and data inputs are in a stable state before testing.
- Clarify what constitutes a “model” in your context. This could be a Python script using a solver API directly (e.g., `gurobipy`), an .MPS file exported from another application, or another supported format.
- Identify any solver-specific modeling techniques already in use, so the benchmarking process can take them into account appropriately. For example, if your model was previously simplified to work around the limitations of another solver, consider testing a less simplified version of the original problem.

Suggested reading: [How does a Gurobi Trial and Evaluation work?](#)

c. Define the benchmarking criteria

While solve speed is often top of mind, the most relevant performance metrics will depend on your specific application and how the model is used in practice. For example, in real-time or near-real-time settings, solve time to reach an acceptable MIP Gap quickly may be critical. In contrast, long-term planning exercises may tolerate longer solve times if the solver can consistently find proven optimal solutions. Below are some additional criteria to consider.

SUGGESTED PERFORMANCE CRITERIA

- Solve time (wall-clock)
- Model build time (wall-clock)
- Solution quality (objective value, gap, violations, tolerances)
- Stability (consistency of results across runs)
- Scalability (ability to handle larger model versions)
- Compute resource requirements (threads, memory)

SUGGESTED MODELING CRITERIA

- API availability and ease of use
- Ease of integration (API availability, language support)
- Modeling functionality and supported model types (MIP, MIQP, MINLP, etc.)

Suggested reading: [What to measure during benchmarking?](#)

d. Define the test environment (hardware and solver versions)

For a robust and reliable benchmarking process, it is important to ensure all solvers and settings being compared run on the same hardware and environment, without any separate applications consuming significant resources.

Gurobi offers a variety of licensing options to support different evaluation environments, from laptops and on-prem servers to cloud-based and containerized deployments. To ensure the appropriate license configuration, share a brief description of your intended testing and deployment setup (e.g., laptop, cloud, containers, hardware specifications, etc.) with your account team.

This is especially relevant for legacy models that have been in production for years under older operating conditions. Advances in solver performance may now enable faster solve times and better-quality solutions, without the compromises made in the past.

Suggested reading: [What hardware should I select when running Gurobi?](#) and [How do I choose a Gurobi license type?](#)

e. Define support and ecosystem evaluation criteria

Choosing a solver is not just about performance criteria. It is equally important to consider the maturity of the solver resources and long-term sustainment outlook, including areas like:

- **Documentation:** ease of information access, clarity, completeness
- **Online resources:** community presence, tutorials, learning materials
- **Support responsiveness:** time to first reply, time to resolution, ability to prioritize production-blocking issues
- **Quality of support responses:** completeness, depth, practical value
- **Licensing flexibility:** commercial terms, license models
- **Ease of integration with enterprise workflows:** APIs in various languages, platform compatibility
- **Future proofing:** release cadence, ecosystem investments
- **User base:** access to professionals with verified experience in production environments

GUROBI TIP

Submit an identical support ticket to each vendor during your benchmark and track actual response and resolution times (hours/days) and quality.



SECOND PHASE

Benchmarking.

Benchmarking can be as simple as just running a solver under the default settings, or it can be a complex engagement involving parameter tuning and model adjustments to improve the solver performance as much as possible. Regardless of complexity, benchmarking gives you valuable insight into solver behavior under realistic conditions, laying the groundwork for a confident, informed decision.

Key activities.

a. Finalize metrics for benchmarking criteria

Your definition of success is defined by your evaluation criteria. The solver metrics you prioritize, such as time to optimality, first feasible solution, or solution quality within a time limit, should reflect your specific use case. Design your benchmarking tests accordingly. The table below outlines common success metrics used in solver evaluations, and there are others listed in this article: [What to measure during benchmarking?](#)

Metric	Description	Use Case	Test criteria to consider
Time to optimality	Measure solver time (not including model build time) until target MIPGap is reached	When proving global optimality is important	What MIPGap represents 'optimality' for your use case? In some case 1% may be acceptable, or you may need .01%.
Solution quality after fixed timespan	Using a fixed TimeLimit as termination criteria, measure the quality of the solution after the TimeLimit is reached	When quality matters more than optimality, and when there is a limit on the time a solver can spend on the solution	Consider multiple quality metrics when evaluating solutions, like constraint violations, MIPGap , and the ObjVal
Time to first solution	Measure solver time needed to find any feasible solution to the model	When finding any solution as fast as possible is the main goal	Use the SolutionLimit parameter to target a single feasible solution
Time to target value	Measure solver time needed to reach a particular Objective value	When proving optimality is not necessary, but there is a certain quality target to reach	Specify the target objective using the BestObjStop parameter

Hardware benchmarking is another important consideration, particularly when deploying optimization models in cloud environments. The goal is to determine the minimum hardware (e.g., CPU cores, memory) required to achieve acceptable runtime performance. This helps reduce compute costs while maintaining solution quality.

Suggested reading: [How many cores does my model need?](#)

NOTE ON STABILITY (FOR ALL SUCCESS METRICS)

Solver performance can vary from run to run, even on the same model and machine. While Gurobi is [designed to be deterministic](#), factors such as [hardware conditions](#), background processes (e.g., running on a laptop), or inconsistent solver settings can impact [performance variability](#).

For this reason, we recommend measuring average runtime and runtime variance across multiple random [seeds](#) to obtain reliable comparisons. This approach provides a more meaningful view of solver behavior and helps reduce noise in benchmark results.

Commercial solvers apply tolerances for feasibility, integrality, and optimality. If you are comparing multiple solvers, it is important to ensure that these tolerance settings are equivalent. While default values are often similar across

solvers, this is not guaranteed, and tolerances are sometimes modified to improve performance while still producing acceptable solution quality.

b. Complete baseline tests

Testing with the default settings is a natural starting point. We will call this the Baseline Test. We recommend the following:

- Start by running your model with the solver's default setting (e.g., Gurobi's default parameter values), except for any custom termination criteria relevant to your use case. Note that if you are using a third-party **modeling** framework, some Gurobi parameters may be modified without your knowledge. You can always export .MPS files directly and test in the Gurobi python API, or export .prm files to verify what parameter settings Gurobi was using.
- Run the model using at least three different random seeds and record the average results across your chosen metric(s). Also note the variance between runs. If the variation is larger than expected, consider testing with additional seeds to investigate further.
- If you are testing multiple datasets, run the model with each dataset independently and record the results. (For each dataset, you should still test multiple seeds.)
- If scalability is part of your evaluation criteria, prepare small, medium, and large versions of your model and establish a separate baseline for each.

These baseline results will serve as the reference point for tuning (if needed). We recommend sharing them with your Gurobi team for feedback and alignment to determine if the next step is necessary.

c. Optional: tune parameters for additional performance

In many cases, Gurobi's default settings provide the best performance. However, certain models can benefit from customized solver settings to unlock their full potential. Depending on your model type, certain components of the Gurobi Optimizer algorithm may deserve closer attention. For example:

- Does your MIP benefit from the NoRel heuristic?
- Does your LP require running crossover?
- Does your MINLP have better performance with our exact solver or is there some benefit to using a piecewise linear approximation?

When comparing solvers, try to apply a consistent level of tuning effort across all tools. Avoid over-optimizing one solver while leaving others in their default configurations.

Gurobi offers several ways to fine-tune solver performance:

- Nearly 200 **solver parameters** can be adjusted for your model characteristics.
- We recommend consulting with our support team before using the Automatic Parameter Tuning Tool, which can require significant time and hardware resources. Our team can suggest targeted parameter settings based on your model structure and objectives.
- Gurobi's Automatic Parameter Tuning Tool explores parameter settings to optimize performance.

Suggested reading: [What is parameter tuning?](#)

If tuning alone does not yield satisfactory results, you may want to consider alternate model formulations. Model reformulation can significantly improve solver performance by addressing issues in how the problem is structured (e.g., variable scaling, symmetry, redundant constraints, weak vs. tight formulations, etc.).

d. Document and log test results

Capturing results consistently becomes especially important when testing multiple models, solver configurations, hardware options, or parameter sets. Your final reporting format may vary based on the evaluation metrics you have selected, and the number of tests performed. To support clear comparisons, use a structured format, like the sample table provided below. This example is for a case where time to optimality is the main benchmarking metric. If you tested multiple model sizes, then each model would have its own table as well.

Core metric: time to optimality

Solver	# of Seeds Tested	Average Solve Time (wall-clock)	Range of Solve Time (wall-clock)	# of runs where optimality is reached
Gurobi	#	Time in seconds/ minutes	Variance measure or Stable/Unstable	Fraction of # of seeds tested
Alternate Solver	#	Time in seconds/ minutes	Variance measure or Stable/Unstable	Fraction of # of seeds tested

Note: If there are some seeds with significantly different solve times or seeds where optimality is not reached within the Time Limit, consider reviewing your model for issues with numerical stability and scaling.

If you are conducting hardware benchmarking to determine compute resource requirements, a table like this may be helpful:

Solver	# of Seeds Tested	Threads = 1	Threads = 2	Threads = 8
Gurobi	#	Average Time $\pm$ variance	Average Time $\pm$ variance	Average Time $\pm$ variance
Alternate Solver	#	Average Time $\pm$ variance	Average Time $\pm$ variance	Average Time $\pm$ variance

You may also want to include a row in your results tables for any existing 'status quo' approach, even if that approach does not use optimization. Comparing your current heuristic or manual method with the results from a mathematical solver can offer a compelling and intuitive illustration of the value of provably optimal solutions.



THIRD PHASE

Decision.

During the decision phase, you will synthesize the insights from your benchmarking activities and make a final determination about which solver best meets your organization's needs. This includes analyzing test results, weighing technical and business considerations, and preparing for internal alignment and adoption. If you choose to move forward with Gurobi, this is also the time to begin commercial discussions with your account manager and plan for a successful rollout.

Key activities.

a. Analyze test results

If you have already been building out tables with your benchmarking results, it should now be easy to identify which solver performed best under specific conditions. The Gurobi team is available to help interpret results and review log files if you would like to explore performance details more deeply.

GUROBI TIP

With `gurobi-logtools`, you can extract information from Gurobi log files and generate pandas DataFrames or Excel worksheets for further processing. We also offer a `plot` method which combines the power of interactive dashboards through `ipywidgets` and plotting functions from `plotly.express`, making it easy to explore your data and results.

b. Make a decision

The final step in your evaluation process is to make a clear, confident decision about which solver best meets your organization's technical and business needs. This decision should be guided by a thorough review of your benchmarking results, ecosystem criteria, and the input of key stakeholders.

While solver performance is critical, other factors, such as ease of integration, quality of support, and total cost of ownership, often play an equally important role in successful deployment.

Budget will likely play a role in your final decision. As you evaluate licensing options, it is important to consider the broader impact of solver efficiency and reliability. A solver that delivers faster runtimes, better-quality solutions, or more robust tooling can lead to shorter development cycles, fewer engineering workarounds, and reduced operational risk, ultimately providing long-term value that outweighs upfront costs.

c. Begin a commercial discussion with the solver vendor

Once you have identified your preferred solver, the next step is to begin a commercial discussion. With Gurobi, this typically includes topics such as licensing models and potential long-term partnership opportunities. This commercial conversation is a key part of finalizing your evaluation and preparing for successful adoption, including the internal communication and justification needed for stakeholders such as IT, procurement, and executive sponsors.

Gurobi can help you navigate these internal discussions by providing tailored materials, technical context, and support to ensure that your organization is fully aligned and set up for long-term success.

Conclusion.

The evaluation and benchmarking of an optimization solver involve understanding performance in the context of your real-world needs. If you have followed the structured process outlined in this guide, you have laid the groundwork for a confident and informed decision. To summarize, as part of a solver evaluation:

- You defined success metrics that matter to your business, whether that is solution quality, reliability, or reproducibility, not just raw speed.
- You documented solver configurations and tuning decisions.
- You ensured consistency in test conditions (hardware and solver settings) across all solvers.
- You evaluated both solver performance and the support ecosystem.

With those foundations in place, you are well positioned to select the solver that best meets your technical and business goals.

Thank you for including Gurobi in your evaluation. We are here to support you, whether you are just starting to explore optimization or preparing for production deployment, and we are committed to helping you get the full value of mathematical optimization.



gurobi.com/free-trial