GNBG-III Competition Protocols (MATLAB Harness)

Participant Guide

Scope

This document describes the **exact evaluation protocol** implemented by the provided MATLAB harness `GNBG_III_CompetitionHarness.m` and the required participant algorithm interface. Participants should implement their method by **replacing the internal logic** inside the provided algorithm template (e.g., `runAlgorithmTemplate.m`) while keeping the interface unchanged.

Benchmark suite

The harness evaluates **24 GNBG-III problems** (files `F1_...` to `F24_...` in `GNBG_III_Benchmarks/`). Each problem loads a struct `GNBG` containing (at minimum):

- `GNBG.Dimension` (problem dimension D)
- `GNBG.MaxEvals` (evaluation budget; competition standard is **500,000**)
- `GNBG.MinCoordinate`, `GNBG.MaxCoordinate` (bounds; scalar or length-D)
- `GNBG.OptimumValue` (known optimum objective value $f^\star$)

Special cases in `fitness.m`.

- **F23 (Noisy):** if `GNBG.NoiseLevel` exists, the returned value is perturbed by Gaussian noise.
- **F24 (Dynamic):** if `GNBG.DynamicShift` and `GNBG.DynamicPeriod` exist, the landscape is shifted at function-evaluation counts that are multiples of `DynamicPeriod` (after FE > 0). This happens *inside* `fitness`.

Run setup and randomness

For each problem F_i, the harness performs `RunNumber` **independent runs**. A deterministic seed is set per (problem, run) as:

`rng(100000 + 1000*i + run).`

This ensures full reproducibility as long as the harness is unchanged and your algorithm uses MATLAB's RNG.

Evaluation budget and reporting points

Each run is capped at `GNBG.MaxEvals` evaluations. The harness reports error at the following fixed evaluation points:

`evalPoints` $= \{10{,}000, 50{,}000, 100{,}000, 150{,}000, 200{,}000, 250{,}000, 300{,}000, 350{,}000, 400{,}000, 450{,}000, 500{,}000\}.$

Additionally, multi-target metrics use the thresholds:

`TargetThresholds` $= \{10^{-1}, 10^{-3}, 10^{-5}, 10^{-8}\}.$

Participant algorithm interface (required)

Your algorithm is called once per run as:

```
[BestHistory, BestValue, BestPosition, GNBG, AcceptanceReachPoint] = ...
    AlgorithmHandle(GNBG, AlgorithmParams);

% Optional 6th output:
[BestHistory, BestValue, BestPosition, GNBG, AcceptanceReachPoint, Extra] = ...
    AlgorithmHandle(GNBG, AlgorithmParams);
```

Inputs

- `GNBG`: benchmark struct loaded from the `.mat` file.
- `AlgorithmParams`: a struct containing any algorithm hyperparameters you need (set in the harness).

Outputs

- `BestHistory`: **row vector** of length `GNBG.MaxEvals` storing the **best-so-far objective value** after each function evaluation (FE), i.e., `BestHistory(fe)` is the best value observed up to FE = `fe`. If you output a shorter vector, the harness pads it with the final value; if longer, it truncates.
- `BestValue`: final best objective value.
- `BestPosition`: final best decision vector (size $1 \times D$).
- `GNBG`: updated struct (notably `GNBG.FE` is advanced by `fitness` calls).
- `AcceptanceReachPoint`: FE index at which $|f_{\text{best}} - f^{\star}| \leq 10^{-8}$ is **first reached**, or `Inf` if never reached (your template computes this from `BestHistory`).
- `Extra` (optional): struct for additional diagnostics. The harness reads: `Extra.DiversityHistory` (up to 50 values), `Extra.ImprovementCount`, `Extra.StagnationPeriods`.

How to perform evaluations (mandatory)

All evaluations **must** be performed via:

$$[\texttt{f}, \texttt{GNBG}] = \texttt{fitness(X, GNBG)}.$$

- `X` may be a single solution ($1 \times D$) or a batch ($N \times D$).
- The function increments `GNBG.FE` internally **by 1 per evaluated solution**.
- You must **not exceed** `GNBG.MaxEvals`. When using batch evaluation, ensure that `GNBG.FE + N` does not overrun the remaining budget.

What the harness computes (per run)

Let $f_{\text{best}}(\text{FE})$ be the best-so-far objective value at evaluation count FE, and $f^{\star}$ be `GNBG.OptimumValue`. The harness forms the **error curve**:

$$e(\text{FE}) = |f_{\text{best}}(\text{FE}) - f^{\star}| \,.$$

BestHistory normalization

Before computing metrics, the harness:

- reshapes `BestHistory` to a row vector,
- pads/truncates it to length `MaxEvals`, and
- enforces monotonicity: if `BestHistory(fe) > BestHistory(fe-1)`, it is overwritten by `BestHistory(fe-1)`.

Therefore, even if your internal tracking is imperfect, the harness evaluates a non-increasing best-so-far trace.

Error at evaluation points

For each evaluation point $p \in$ `evalPoints`, the harness records:

$$\texttt{ErrorAtPoints}(p) = e(p).$$

FE-to-target (multi-target)

For each threshold $\tau \in \{10^{-1}, 10^{-3}, 10^{-5}, 10^{-8}\}$, the harness records:

$$\texttt{FEto\_}\tau = \min\{\mathrm{FE} : e(\mathrm{FE}) \le \tau\},$$

and stores `NaN` if never reached within the budget.

Success flags

For each target τ, a run is a **success** if it reaches the target within budget (i.e., `FEto_tau` is finite). For the strictest target (10^{-8}), the harness also stores:

- `AcceptancePoints(run)` = `AcceptanceReachPoint` (finite) or `Inf` (failure),
- `SuccessRate` (percentage of runs with finite `AcceptancePoints`).

Expected Running Time (ERT)

For each target τ, the harness computes:

- Let FE_r be the FE-to-target for run r (NaN if failure).
- Replace failures by `MaxEvals`: $\widehat{\mathrm{FE}}_r = \mathrm{FE}_r$ if success else `MaxEvals`.
- Then
$$\texttt{ERT}(\tau) = \frac{\sum_{r=1}^{R} \widehat{\mathrm{FE}}_r}{\#\{\text{successful runs}\}},$$
and `ERT` is set to `Inf` if there are **zero** successful runs.

Aggregated statistics (per problem)

For each problem, the harness stores per-evaluation-point summary statistics across runs:

$$\texttt{MeanErrors},\ \texttt{StdErrors},\ \texttt{MedianErrors}.$$

It also stores key end-budget summaries (notably at 500K): mean, median, std, min, and max errors, plus success rates and ERTs.

Outputs and submission artifacts

After all problems/runs finish, the harness writes:

- **MAT file:** `GNBG_III_DetailedResults_<AlgorithmName>.mat`
  Contains `Results` (all raw/aggregated arrays), `AlgorithmParams`, `evalPoints`, `TargetThresholds`, and `AlgorithmName`.
- **CSV file:** `GNBG_III_Detailed_Results_<AlgorithmName>.csv`
  Contains **one row per (problem, run)** with:
  - `Algorithm`, `Problem`, `Run`
  - `Error_10K` ... `Error_500K`
  - `Acceptance_FE_1e-8`, `Success_1e-8`
  - For each target τ: `FE_to_`τ and `Success_`τ

Implementation checklist (what to do / avoid)

- **Do not change** the harness logic for evaluation, targets, or statistics.
- Ensure all function evaluations go through `fitness` and **respect the budget**.
- Keep `BestHistory` as **best-so-far per FE**; update it *for every evaluation consumed.*
- Always return `AcceptanceReachPoint` as specified (or `Inf` if never reached).
- Boundary handling is participant-controlled (e.g., clipping, reflection, resampling), but must respect bounds.
- You may populate `Extra` with diagnostics; missing fields are allowed.