

# スペースバルーン軌道シミュレータ（JMA GSM） 開発計画書

版: v0.1（初版ドラフト） / 最終更新: 2026-01-30

## 目的

- ・気象庁 JMA High-Resolution GSM（GRIB2）を用いて、日本域スペースバルーン軌道シミュレーションの精度・運用性を向上する。
- ・落下地点を点ではなく確率領域（例: 50%/90%）として提示する（バルーン・パラメータ不確実性によるアンサンブル）。

## 1. 計画の概要

本計画は、Tawhiriの設計思想（データ取得 内部形式 4D補間 物理モデル RK4積分）を参考にしつつ、日本での用途に最適化して、JMA GSMを入力とする独自シミュレータを段階的に構築する。

完成の定義（使える状態）

- 1) 決定論予測: 軌道と落下地点が出る。
- 2) パラメータ・アンサンブル: N本の軌道と着地点点群が出る。
- 3) 確率領域: 点群から 50%/90% 領域（GeoJSON/KML等）が出る。
- 4) 運用: 予報更新時に前処理し、予測要求では高速に参照できる。

## 2. 粒度・更新頻度

毎回の会話で更新（最優先）

- ・「現在のステータス」 / 「次のアクション（最大3つ）」 / 「ブロッカー」

作業単位で更新

- ・フェーズ別ステータス、主要タスク完了チェック、リスク更新

仕様確定時に更新

- ・インターフェース仕様、データセット仕様、運用要件

## 3. 現在のステータス

意思決定済み

- ・気象データは JMA High-Resolution GSM を第一候補として採用（上層 1000 – 10hPa を含む）。
- ・Tawhiriを直接改造するより、Tawhiriの設計思想を参考に独自実装で進める。
- ・出力は「単一路線」 + 「アンサンブルの確率領域」を標準にする。

進捗

- ・設計方針と計画書テンプレート作成（本書）
- ・参照資料の一次収集（JMA GSM、Tawhiri DeepWiki）

未解決の重要事項（Decision Pending）

- ・内部保存形式: memmap（Tawhiri型） vs zarr（圧縮チャンク）
- ・高度の基準: MSLに統一するか / DEMを必須にするか
- ・インターフェース: まずCLIのみ 後でAPI、または最初からAPI
- ・領域推定手法: 楕円近似 HDR等高線へ段階導入

## 4. 開発フェーズ（マイルストーン）

フェーズ	目的	状態
A	JMA GSM 取得・デコード・保存	Not started
B	風場サンブラ（補間）	Not started
C	物理モデル（合成）	Not started
D	RK4ソルバ	Not started
E	アンサンブル予測	Not started
F	確率領域推定	Not started
G	インターフェース/運用	Not started
H	検証・性能	Not started

## フェーズ別詳細（成果物と完了条件）

- A 成果物: `meteo.fetch / meteo.decode / meteo.store`  
完了条件: 1runのU/V/Z ( 1000 – 10hPa ) を取り込み、再利用できる形で保存できる。
- B 成果物: `meteo.windfield`  
完了条件: (t,lat,lon,alt) (u,v) を単点/バッチで返せる。
- C 成果物: `physics.models / physics.termination`  
完了条件: 上昇 バースト 降下 着地 の2ステージが組める。
- D 成果物: `solver.rk4 / solver.solve`  
完了条件: dt固定RK4 + 終了点二分探索で軌道が出る。
- E 成果物: `ensemble.sampler / ensemble.runner`  
完了条件: N本の軌道と着地点点群が出る。
- F 成果物: `outputs.regions`  
完了条件: 50%/90%領域を出せる（最初は楕円、後でHDR等高線）。
- G 成果物: `cli ( 必要ならapi ) / ops`  
完了条件: 手元またはサーバで予測要求を回せる（キャッシュ含む）。
- H 成果物: `tests / benchmarks / validation`  
完了条件: バックテストと性能測定が自動化され、改善が追える。

## 5. 実装レベルの計画（要点）

### フェーズA（ingestion）

Run探索（latest解決） / ダウンロード（再試行・欠損検知） / GRIB2デコード（U/V/Z抽出） / 内部保存（memmap or zarr） / メタデータ管理

### フェーズB（補間）

時間・緯度・経度の線形補間 + 経度wrap / 鉛直はZを使った二分探索 + 線形補間 / 範囲外のwarning設計 / batch APIを先に確定

### フェーズC-D（物理 + RK4）

モデル合成（速度は足し算、終了条件はOR） / RK4 + 終了点二分探索 / 上昇 降下のステージ連結

### フェーズE-F（アンサンブル + 領域）

パラメータ分布（seed管理） / N軌道の効率実行（可能ならバッチ化） / 領域推定は楕円 HDR等高線へ段階導入

## 6. 成果物（コード/モジュール）

想定リポジトリ構成（Python）：

- meteo/: run探索・取得・デコード・保存・風場補間
- physics/: 上昇/降下/風モデル、終了条件
- solver/: RK4、ステージ連結、終了点二分探索
- ensemble/: サンプラ、アンサンブル実行
- outputs/: JSON/KML/GeoJSON、領域推定
- cli/（必要ならapi/）
- tests/、benchmarks/、docs/

## 7. 外部情報源（一次）

- JMA High-Resolution GSM: Product Information / Tutorial
- Tawhiri DeepWiki: Architecture / Wind Data / Interpolation / Models / Solver
- tawhiri-downloader-container DeepWiki: Getting Started

### URL（コピペ用）

DeepWiki：

<https://deepwiki.com/projecthorus/tawhiri-downloader-container/2-getting-started>  
<https://deepwiki.com/cuspaceflight/tawhiri/1.1-system-architecture>  
<https://deepwiki.com/cuspaceflight/tawhiri/2.1-wind-data-management>  
<https://deepwiki.com/cuspaceflight/tawhiri/2.2-wind-interpolation-system>  
<https://deepwiki.com/cuspaceflight/tawhiri/2.3-flight-physics-models>  
<https://deepwiki.com/cuspaceflight/tawhiri/2.4-numerical-integration-solver>

JMA GSM：

<https://www.wis-jma.go.jp/cms/gsm/>  
[https://www.wis-jma.go.jp/cms/gsm/product\\_information.html](https://www.wis-jma.go.jp/cms/gsm/product_information.html)  
<https://www.wis-jma.go.jp/cms/gsm/tutorial.html>

## 8. 今後の見通し（次のアクション）

直近（最大3つ）

- 1) フェーズAの最小E2E: 1runのU/V/Zを取得 保存 読み直し
- 2) 内部保存形式（memmap vs zarr）を決定
- 3) WindFieldの単点サンプル（まずはnearestでも可）

リスク

GRIB2デコード差 / run欠損や遅延 / アンサンブル計算量（ batch APIとキャッシュ設計で対処）

### 変更履歴（Changelog）

日付	版	変更内容	変更者
2026-01-30	v0.1	初版ドラフト作成	TBD