# PyForecast

Oil & Gas DCA Auto-Forecasting Tool

Comprehensive Documentation

Version 0.1.0

Generated: January 29, 2026

## Table of Contents

# 1. Project Overview

PyForecast is an automated decline curve analysis (DCA) tool for oil and gas production forecasting. It automatically fits hyperbolic decline curves to historical production data, detects regime changes (refracs, workovers), and exports forecasts in ARIES-compatible formats.

## Key Features

* Automatic hyperbolic decline curve fitting with configurable constraints
* Regime change detection for refracs and workovers
* Recency-weighted fitting to prioritize recent production trends
* Batch processing with parallel execution support
* ARIES AC_FORECAST and AC_ECONOMIC export formats
* Comprehensive data validation and quality checks
* Hindcast validation for forecast accuracy measurement
* Ground truth comparison against expert ARIES projections
* Interactive plots and visualization

## Supported Products

PyForecast supports forecasting for three fluid types:

* Oil (bbl/month)
* Gas (mcf/month)
* Water (bbl/month)

## Dependencies

| | |
|---|---|
| **pandas** | - Data manipulation and CSV I/O |
| **numpy** | - Numerical computations |
| **scipy** | - Curve fitting optimization |
| **plotly** | - Interactive visualization |
| **typer** | - CLI framework |
| **pyyaml** | - Configuration file support |

# 2. Installation & Quick Start

## Installation

Install PyForecast using pip:

```
pip install pyforecast
```

Or install from source:

```
git clone https://github.com/yourorg/pyforecast.git
cd pyforecast
pip install -e .
```

## Quick Start

### 1. Generate a configuration file

```
pyforecast init -o pyforecast.yaml
```

### 2. Process production data

```
pyforecast process production_data.csv -o output/
```

### 3. Review outputs

After processing, the output directory contains:

* forecasts.csv - Fitted parameters and EUR estimates
* ac_economic.csv - ARIES AC_ECONOMIC export format
* validation_report.txt - Data quality issues
* plots/ - Individual well decline curve plots

## Input Data Format

PyForecast accepts CSV files with production data. Required columns include a well identifier (API or PROPNUM) and date/production columns.

```
API,Date,Oil,Gas,Water
42001000010000,2020-01-01,1500,5000,200
42001000010000,2020-02-01,1400,4800,210
```

# 3. Core Concepts

## Decline Curve Analysis (DCA)

Decline curve analysis is a technique for forecasting oil and gas production by fitting mathematical models to historical production trends. The underlying assumption is that production will continue to decline in a predictable pattern based on reservoir physics and well performance.

## Key Parameters

### qi (Initial Rate)

Production rate at the start of decline (bbl/day or mcf/day)

### di (Initial Decline)

Rate of decline at time zero (fraction per month)

### b (Decline Exponent)

Controls curvature: 0=exponential, 0.5=typical, 1=harmonic

### dmin (Terminal Decline)

Minimum annual decline rate (typically 6%)

## Regime Detection

PyForecast automatically detects regime changes (refracs, workovers, artificial lift installations) that cause production to increase. When detected, fitting starts from the most recent regime rather than historical peak production.

Detection criteria (configurable):

* Production increase > 100% (threshold)
* Sustained for 2+ months (sustained_months)
* Evaluated over 6-month windows (window)

## Recency Weighting

By default, PyForecast weights recent data points more heavily than older data using exponential decay. This helps the fit track current well behavior rather than being anchored to historical patterns that may no longer apply.

Default half-life: 12 months (configurable via recency_half_life)

# 4. Decline Curve Models

## Hyperbolic Decline

The general hyperbolic decline equation relates production rate (q) to time (t):

q(t) = qi / (1 + b * di * t)^(1/b)

Where:

* q(t) = production rate at time t
* qi = initial production rate
* di = initial decline rate
* b = decline exponent (0 to ~1.5)

## Special Cases

### Exponential Decline (b = 0)

q(t) = qi * exp(-di * t)

Exponential decline represents constant percentage decline. Common in solution gas drive reservoirs and late-time production.

### Harmonic Decline (b = 1)

q(t) = qi / (1 + di * t)

Harmonic decline represents proportional decline rate. Less common but may occur in gravity drainage or some water drive reservoirs.

## Terminal Decline (dmin)

Hyperbolic decline predicts ever-decreasing decline rates, which is physically unrealistic at late times. PyForecast switches to exponential decline when the instantaneous decline rate reaches dmin (default 6%/year).

## B-Factor Interpretation

| B Range | Description | Typical Reservoir |
|---|---|---|
| 0.0 - 0.3 | Near-exponential | Solution gas drive, depletion |
| 0.3 - 0.6 | Typical hyperbolic | Most unconventional wells |
| 0.6 - 1.0 | High hyperbolic | Multi-phase flow, complex reservoirs |
| 1.0 - 1.5 | Super-harmonic | Transient flow, early time data |

# 5. CLI Reference

## Main Commands

### pyforecast process

Process production data and generate forecasts

### pyforecast init

Generate a default configuration file

### pyforecast analyze-fits

Analyze accumulated fit logs

### pyforecast suggest-params

Get parameter suggestions from learning

### pyforecast calibrate-regime

Calibrate regime detection thresholds

## Process Command Options

```
pyforecast process INPUT_FILE [OPTIONS]
```

| Option | Description | Default |
|---|---|---|
| -o, --output | Output directory | Required |
| -c, --config | Configuration file | Optional |
| -p, --product | Products to forecast (oil,gas,water) | All |
| --plots/--no-plots | Generate individual well plots | True |
| --batch-plot/--no-batch-p | Generate batch overlay plot | True |
| --format | Export format (ac_forecast, ac_economic, | ac_economic |
| --hindcast | Enable hindcast validation | False |
| --log-fits | Enable fit logging | False |
| --residuals | Enable residual analysis | False |
| --ground-truth | ARIES file for comparison | None |
| --gt-months | Months to compare | 60 |
| --gt-plots | Generate comparison plots | False |
| --gt-lazy | Stream ARIES file (low memory) | False |
| --gt-workers | Parallel validation workers | 1 |

# 6. Data Validation

PyForecast includes a comprehensive validation system that checks data quality at multiple stages: input validation, data quality checks, and fit result validation.

## Validation Categories

| Category | Description |
|---|---|
| DATA_FORMAT (IV) | Column, date, and value format issues |
| DATA_QUALITY (DQ) | Gaps, outliers, shut-ins, variability |
| FITTING_PREREQ (FP) | Pre-fit requirements not met |
| FITTING_RESULT (FR) | Post-fit quality concerns |

## Severity Levels

* ERROR - Cannot proceed safely, must resolve
* WARNING - Can proceed with caution, review recommended
* INFO - Informational only, no action required

## Input Validation Codes

| Code | Severity | Description |
|---|---|---|
| IV001 | ERROR | Negative production values |
| IV002 | WARNING | Values exceed threshold |
| IV003 | ERROR | Date parsing failed |
| IV004 | WARNING | Future dates in data |

## Data Quality Codes

| Code | Severity | Description |
|---|---|---|
| DQ001 | WARNING | Data gaps detected (>= threshold months) |
| DQ002 | WARNING | Outliers detected (> sigma threshold) |
| DQ003 | INFO | Shut-in periods detected |
| DQ004 | WARNING | Low data variability (CV < threshold) |

## Fitting Prerequisite Codes

| Code | Severity | Description |
|---|---|---|
| FP001 | ERROR | Insufficient data points |
| FP002 | WARNING | Increasing trend (not declining) |
| FP003 | WARNING | Flat trend (minimal decline) |

## Fitting Result Codes

| Code | Severity | Description |
|---|---|---|
| FR001 | WARN/ERR | Poor fit quality (R-squared < threshold) |
| FR003 | INFO | B-factor at lower bound |
| FR004 | WARNING | B-factor at upper bound |
| FR005 | WARNING | Very high decline rate (>100%/year) |

## Validation Report

A validation_report.txt file is generated in the output directory with detailed information about all validation issues found during processing.

# 7. Refinement & Analysis

The refinement module provides tools for measuring, logging, analyzing, and improving decline curve fit quality. All features are disabled by default.

## Hindcast Validation

Hindcast validation splits historical data into training and holdout periods, fits on training data, and measures prediction accuracy on the holdout.

Metrics computed:

* MAPE - Mean Absolute Percentage Error (good: < 30%)
* Correlation - Pearson correlation coefficient (good: > 0.7)
* Bias - Systematic over/under prediction (good: < 0.3)

```
pyforecast process data.csv --hindcast -o output/
```

## Fit Logging

Fit logging persists all fit parameters and metrics to SQLite storage for analysis and learning across projects.

Default storage: ~/.pyforecast/fit_logs.db

```
pyforecast process data.csv --log-fits -o output/
pyforecast analyze-fits ~/.pyforecast/fit_logs.db
```

## Residual Analysis

Residual analysis detects systematic fit errors that may not be apparent from R-squared alone, including autocorrelation and early/late bias patterns.

Diagnostics computed:

* Durbin-Watson statistic (ideal: ~2.0)
* Early bias (error in first half)
* Late bias (error in second half)
* Lag-1 autocorrelation

```
pyforecast process data.csv --residuals -o output/
```

## Parameter Learning

Parameter learning analyzes accumulated fit logs to suggest optimal fitting parameters for different basins and formations.

```
pyforecast suggest-params -p oil --basin Permian
```

# 8. Ground Truth Comparison

Ground truth comparison measures how well pyforecast's auto-fitted decline curves match expert/approved ARIES projections.

## Use Case

When you have existing ARIES forecasts created by reservoir engineers, you can compare pyforecast's automated fits against these 'ground truth' projections to:

* Validate that auto-fitting produces reasonable results
* Identify wells where manual review may be needed
* Measure overall fitting accuracy across a portfolio

## CLI Usage

```
pyforecast process data.csv --ground-truth aries.csv -o output/

# With comparison plots
pyforecast process data.csv --ground-truth aries.csv --gt-plots -o output/

# Large file support
pyforecast process data.csv --ground-truth large.csv --gt-lazy --gt-workers 4 -o output/
```

## Metrics Computed

| Metric | Description | Good Value |
|---|---|---|
| MAPE | Mean Absolute Percentage Error | < 20% |
| Correlation | Pearson correlation of curves | > 0.95 |
| Cumulative Diff | Total production difference | < 15% |
| B-Factor Diff | Absolute b-factor difference | < 0.3 |

## Match Quality Grades

| Grade | Description |
|---|---|
| A | Excellent match - all metrics well within thresholds |
| B | Good match - minor deviations |
| C | Fair match - some significant differences |
| D | Poor match - review recommended |
| X | Insufficient data - MAPE could not be calculated |

## ARIES Expression Format

PyForecast parses ARIES AC_ECONOMIC expression format:

```
"{Qi} X {unit} {Dmin%} {type} B/{b} {Di%}"

Example: "1000 X B/M 6 EXP B/0.50 8.5"
```

Supported units:

* B/M - barrels/month (oil)
* B/D - barrels/day (oil)
* M/M - mcf/month (gas)
* M/D - mcf/day (gas)

## Advanced Features

### Rate Validation

Forecast arrays are automatically validated for NaN, infinite, and negative values. Issues are logged as warnings and values are cleaned before metric calculation.

### MAPE Edge Case Handling

When fewer than 3 valid data points exist above the rate threshold, MAPE returns None and match_grade returns 'X' (insufficient data).

### Identifier Mismatch Logging

The validate_batch() method tracks wells that exist in only one dataset, helping diagnose ID normalization issues between pyforecast and ARIES data.

### Comparison Plots

Generate overlay plots showing ARIES vs pyforecast curves with metrics annotation. Plots are saved to output/ground_truth_plots/ directory.

### Lazy Loading

For large ARIES files (10,000+ wells), use --gt-lazy flag to stream data from disk instead of loading into memory. Uses constant memory regardless of file size.

### Parallel Validation

For large batches, use --gt-workers N to enable parallel validation with ThreadPoolExecutor. Default is 1 (sequential).

# 9. Configuration Reference

PyForecast uses YAML configuration files. Generate a default with:

```
pyforecast init -o pyforecast.yaml
```

## Product Configuration

```
oil:
  b_min: 0.01      # Minimum b-factor
  b_max: 1.5       # Maximum b-factor
  dmin: 0.06       # Terminal decline (annual)
```

## Regime Detection

```
regime:
  threshold: 1.0          # Min increase to trigger (1.0 = 100%)
  window: 6               # Months of trend data
  sustained_months: 2     # Months elevation must persist
```

## Fitting Parameters

```
fitting:
  recency_half_life: 12.0  # Half-life for data weighting
  min_points: 6            # Minimum months required
```

## Output Configuration

```
output:
  products: [oil, gas, water]
  plots: true
  batch_plot: true
  format: ac_economic
```

## Validation Configuration

```
validation:
  max_oil_rate: 50000
  max_gas_rate: 500000
  gap_threshold_months: 2
  outlier_sigma: 3.0
  min_r_squared: 0.5
  strict_mode: false
```

## Refinement Configuration

```yaml
refinement:
  enable_logging: false
  log_storage: sqlite
  enable_hindcast: false
  hindcast_holdout_months: 6
  min_training_months: 12
  enable_residual_analysis: false
  ground_truth_file: null
  ground_truth_months: 60
  ground_truth_lazy: false
  ground_truth_workers: 1
```

# 10. Module Reference

## Package Structure

```
pyforecast/
  core/
    models.py         # Decline curve models
    fitting.py        # Curve fitting logic
    regime_detection.py  # Regime change detection
    selection.py      # Model selection
  validation/
    input_validator.py
    data_quality_validator.py
    fitting_validator.py
  refinement/
    hindcast.py       # Hindcast validation
    fit_logger.py     # Fit logging
    residual_analysis.py
    parameter_learning.py
    ground_truth.py   # Ground truth comparison
    plotting.py       # Comparison plots
    schemas.py        # Data classes
    storage.py        # SQLite storage
  import_/
    aries_forecast.py # ARIES file parser
  export/
    ac_forecast.py    # AC_FORECAST export
    ac_economic.py    # AC_ECONOMIC export
  batch/
    processor.py      # Batch processing
  cli/
    commands.py       # CLI commands
  config.py           # Configuration
```

## Key Classes

| Class | Description |
|---|---|
| HyperbolicModel | Decline curve model with qi, di, b, dmin parameters |
| DeclineFitter | Fits decline curves to production data |
| RegimeDetector | Detects production regime changes |
| HindcastValidator | Validates forecasts via backtesting |
| FitLogger | Logs fit parameters to storage |
| GroundTruthValidator | Compares against ARIES projections |
| AriesForecastImporter | Parses ARIES AC_ECONOMIC files |
| InputValidator | Validates input data format/values |
| DataQualityValidator | Checks for gaps, outliers, etc. |