Introduction to Model Fitting and Calibration

Modelling for Pandemic Preparedness and Response Modular Shortcourse, 2026

James Mba Azam, PhD

2026-08-28

Learning Objectives

By the end of this lecture, you will be able to:

- Understand the fundamental concepts of model fitting and calibration
- Apply least squares estimation to compartmental models
- Implement maximum likelihood estimation for epidemic models
- Compare the strengths and weaknesses of different fitting methods
- Recognize when to use advanced methods like MCMC and particle filtering
- Troubleshoot common fitting problems

Introduction & motivation

Why model fitting matters

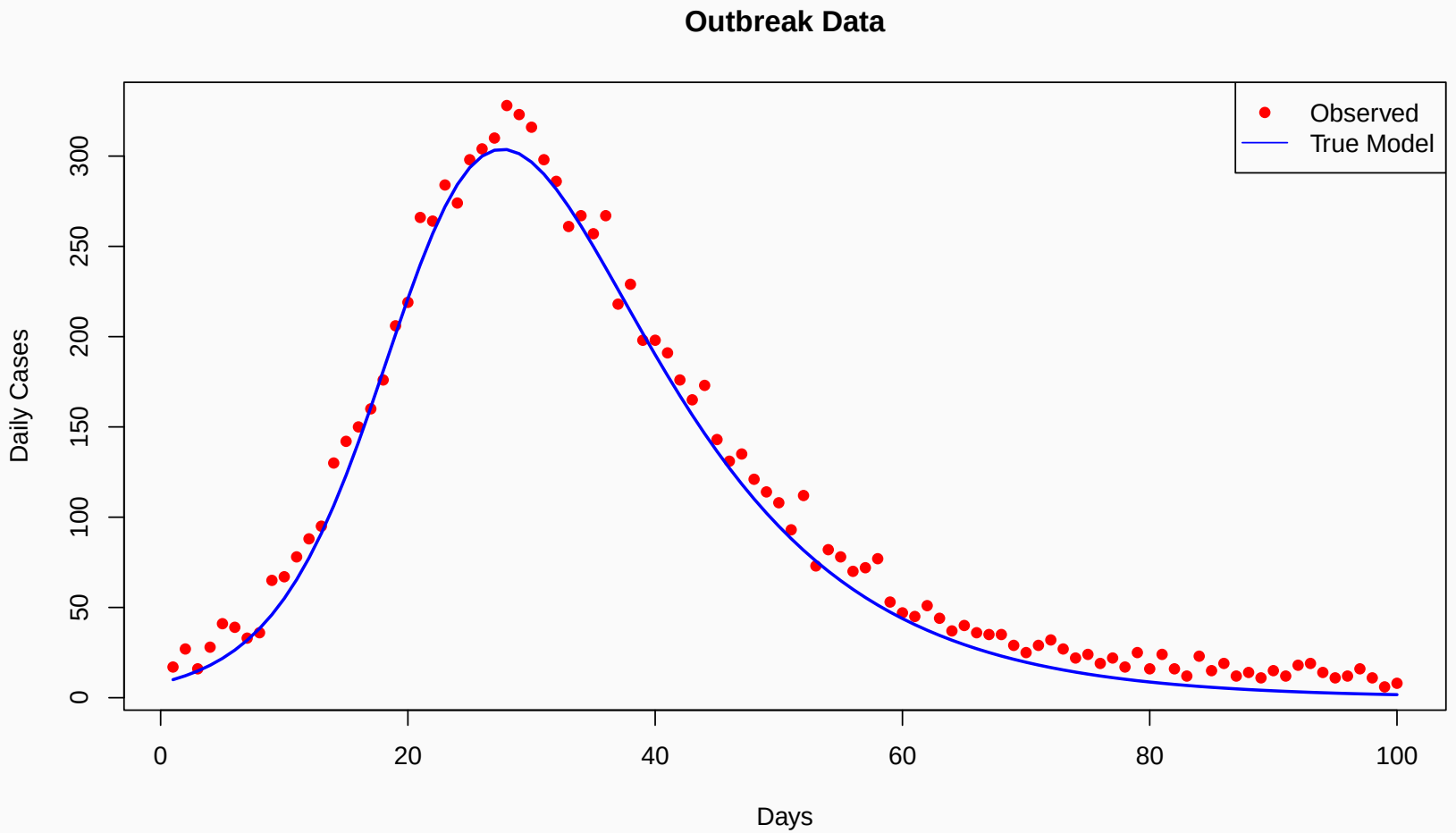
The challenge

- We have mathematical models (e.g., SIR, SEIR)
- We have real-world data (case counts, hospitalizations)
- **How do we connect them?**

The goal

- Find parameter values that make our model predictions match observed data
- Quantify uncertainty in our estimates
- Make reliable predictions and policy recommendations

Example: outbreak data



Question: How do we estimate β and γ from this noisy data?

Types of fitting methods

Deterministic methods

- **Least Squares**
- **Maximum Likelihood Estimation (MLE)**

Stochastic methods

- **Markov Chain Monte Carlo (MCMC)**
- **Sequential Monte Carlo (SMC)**
- **Particle MCMC (pMCMC)**
- **Approximate Bayesian Computation (ABC)**

Today's Focus: Least Squares and MLE as foundations

Conceptual foundations

What is model fitting?

Definition: The process of finding parameter values that make a mathematical model's predictions as close as possible to observed data.

Mathematical formulation

$$\hat{\theta} = \arg\min_{\theta} \mathcal{L}(\theta, \mathbf{y})$$

Where:

- $\theta =$ parameter vector (e.g., β, γ)
- $\mathbf{y} =$ observed data
- $\mathcal{L} =$ loss/objective function

The SIR model as our example

Differential Equations:

$$\frac{dS}{dt} = -\beta SI \tag{1}$$

$$\frac{dI}{dt} = \beta SI - \gamma I \tag{2}$$

$$\frac{dR}{dt} = \gamma I \tag{3}$$

Parameters to estimate

- $\beta =$ transmission rate
- $\gamma =$ recovery rate

Key quantities

- $R_0 = \frac{\beta}{\gamma}$ (basic reproduction number)
- Infectious period $= \frac{1}{\gamma}$

Sources of uncertainty

Model uncertainty

- Wrong model structure
- Missing compartments or processes

Parameter uncertainty

- True parameter values unknown
- Multiple parameter sets give similar fits

Observation uncertainty

- Measurement error
- Reporting delays
- Underreporting

Process uncertainty

- Stochasticity in disease transmission
- Environmental variability

The fitting challenge

Identifiability Problem: Multiple parameter combinations can produce similar model outputs

Example:

- High β, high γ
- Low β, low γ

Both might give similar epidemic curves!

Solution: Use additional information (e.g., known infectious period)

Least squares estimation

Least squares: the intuitive approach

Core Idea: Minimize the sum of squared differences between model predictions and observations

Mathematical Formulation:

$$\text{SSE} = \sum_{i=1}^{n} (y_i - f(t_i, \theta))^2$$

Where:

- y_i = observed value at time t_i
- $f(t_i, \theta)$ = model prediction at time t_i
- θ = parameter vector

Why squared errors?

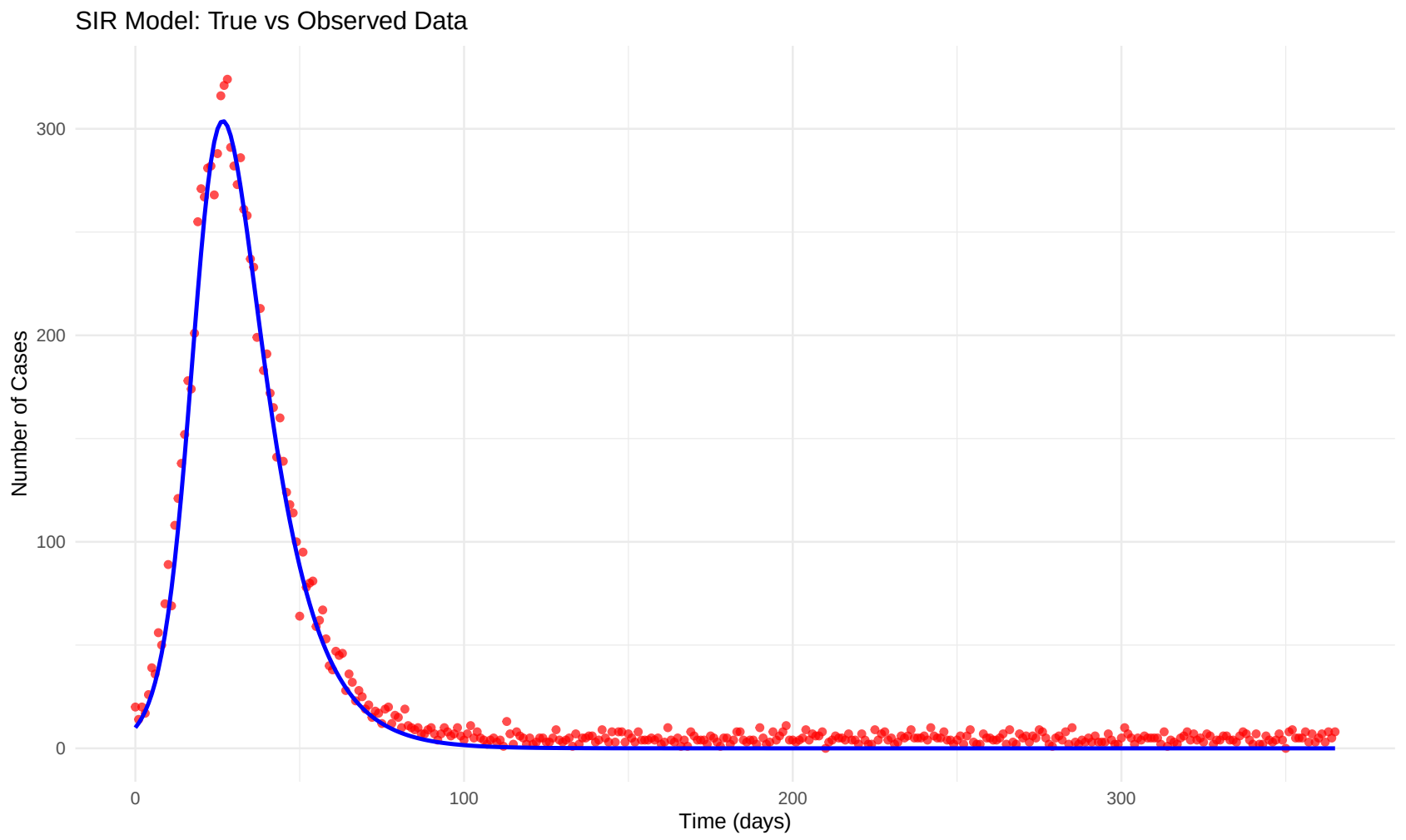
Advantages:

- Penalizes large errors more heavily
- Differentiable (smooth optimization)
- Mathematically tractable
- Gives maximum likelihood estimates when errors are normal distributed

Disadvantages

- Sensitive to outliers
- Assumes constant variance
- No probabilistic interpretation

Least squares example: SIR model



Implementing least squares

```r
# Define objective function
sse_function <- function(params, data) {
  beta <- params[1]
  gamma <- params[2]

  # Simulate model
  out <- ode(
    y = init_conds,
    times = data$time,
    func = sir_model,
    parms = c(beta = beta, gamma = gamma)
  )

  #  Extract and rescale the predicted cases
  predicted <- out[, "I"] * 1000
```

```r
# Prepare data
data <- data.frame(time = times, observed = observed_cases)

# Test different parameter values
test_params <- expand.grid(
  beta = seq(0.1, 0.5, length.out = 10),
  gamma = seq(0.05, 0.2, length.out = 10)
)
```

```r
# Calculate SSE for each combination
for (i in 1:nrow(test_params)) {
  test_params$sse[i] <- sse_function(
    c(
      test_params[i, 1],
      test_params[i, 2]
    ),
    data
  )
}
```

```r
# Find minimum
best_idx <- which.min(test_params$sse)
best_params <- test_params[best_idx, ]
cat("True parameters: Beta =", true_params[1], ", Gamma ="
```

```
True parameters: Beta = 0.3 , Gamma = 0.1
```

```r
cat("Best fit parameters: Beta =", round(best_params$beta,
```

```
Best fit parameters: Beta = 0.322 , Gamma = 0.1
```

Strengths of least squares

Computational Advantages:

- Fast and efficient
- Well-established algorithms
- Easy to implement
- Good for initial parameter estimates

Statistical properties

- Unbiased estimates (under certain conditions)
- Minimum variance among linear unbiased estimators
- Maximum likelihood when errors are normal

Practical benefits

- Intuitive interpretation
- Widely understood
- Good starting point for more complex methods

Limitations of least squares

Statistical Limitations:

- Limited uncertainty quantification
- Assumes constant variance
- Sensitive to outliers
- No probabilistic framework

Practical limitations

Practical Limitations:

- Parameter identifiability issues
- No confidence intervals
- Difficult to compare models
- Assumes measurement error only

R implementation practicals

- Let's turn to the tutorials

Maximum likelihood estimation

Maximum likelihood: the probabilistic approach

Core Idea: Find parameter values that make the observed data most probable

Mathematical Formulation:

$$\hat{\theta} = \arg\max_{\theta} L(\theta) = \arg\max_{\theta} \prod_{i=1}^{n} f(y_i|\theta)$$

Where:

- $L(\theta) =$ likelihood function
- $f(y_i|\theta) =$ probability density of observation i

In Practice: Maximize log-likelihood

$$\hat{\theta} = \arg\max_{\theta} \ell(\theta) = \arg\max_{\theta} \sum_{i=1}^{n} \log f(y_i|\theta)$$

Why maximum likelihood?

Theoretical advantages:

- Principled statistical framework
- Provides uncertainty quantification
- Enables model comparison (AIC, BIC)
- Asymptotically optimal properties

Practical benefits

- Confidence intervals
- Hypothesis testing
- Model selection
- Incorporates different error structures

Choosing a probability distribution

For Count Data (e.g., Cases):

- **Poisson**: $Y_i \sim \text{Poisson}(\lambda_i)$
- **Negative Binomial**: $Y_i \sim \text{NB}(\mu_i, \phi)$

Choosing a probability distribution

For Continuous Data:

- **Normal**: $Y_i \sim N(\mu_i, \sigma^2)$
- **Log-normal**: $\log Y_i \sim N(\log \mu_i, \sigma^2)$

> **Note**
>
> **For Our SIR Example:** We'll use Poisson since we're modeling case counts

MLE implementation: Poisson likelihood

```r
# Define negative log-likelihood function
nll_function <- function(beta, gamma, data) {
  # Simulate model
  out <- ode(y = init_conds, times = data$time,
             func = sir_model, parms = c(beta = beta, gamma
  # Model predictions (scaled to cases)
  predicted <- out[,"I"] * 1000
  # Poisson negative log-likelihood
  nll <- -sum(dpois(data$observed, lambda = predicted, log
  return(nll)
}
```

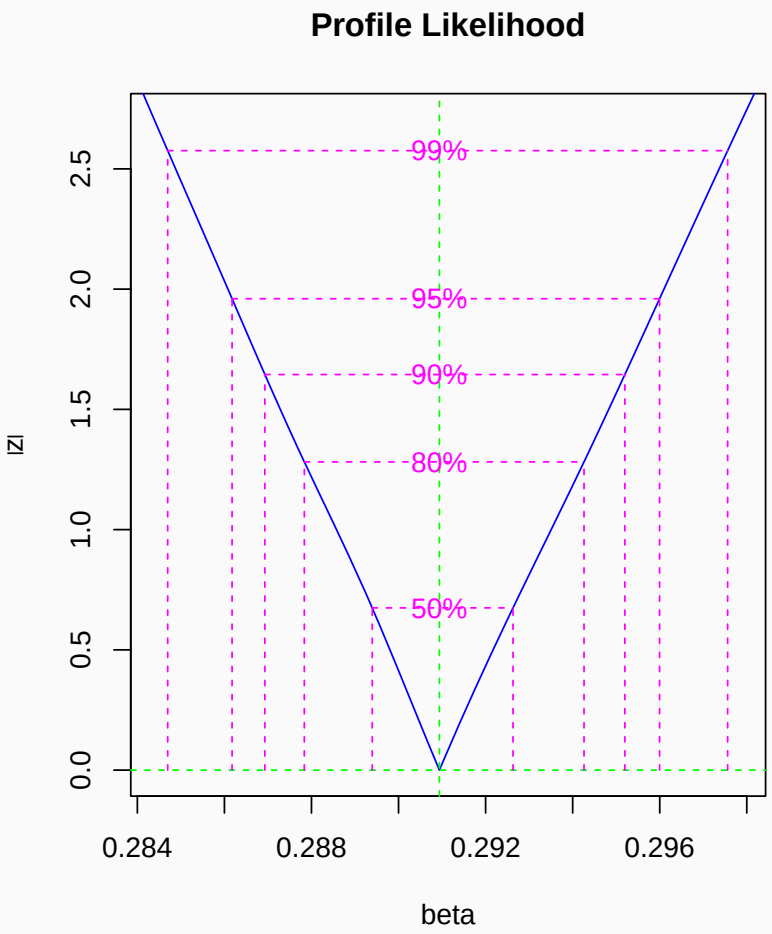
```r
# Use optimization to find MLE
library(bbmle)
fit_mle <- mle2(nll_function,
                start = list(beta = 0.2, gamma = 0.1),
                data = list(data = data),
                method = "L-BFGS-B",
                lower = c(0.01, 0.01),
                upper = c(1.0, 0.5))

# Extract results
mle_params <- coef(fit_mle)
mle_se <- sqrt(diag(vcov(fit_mle)))
```

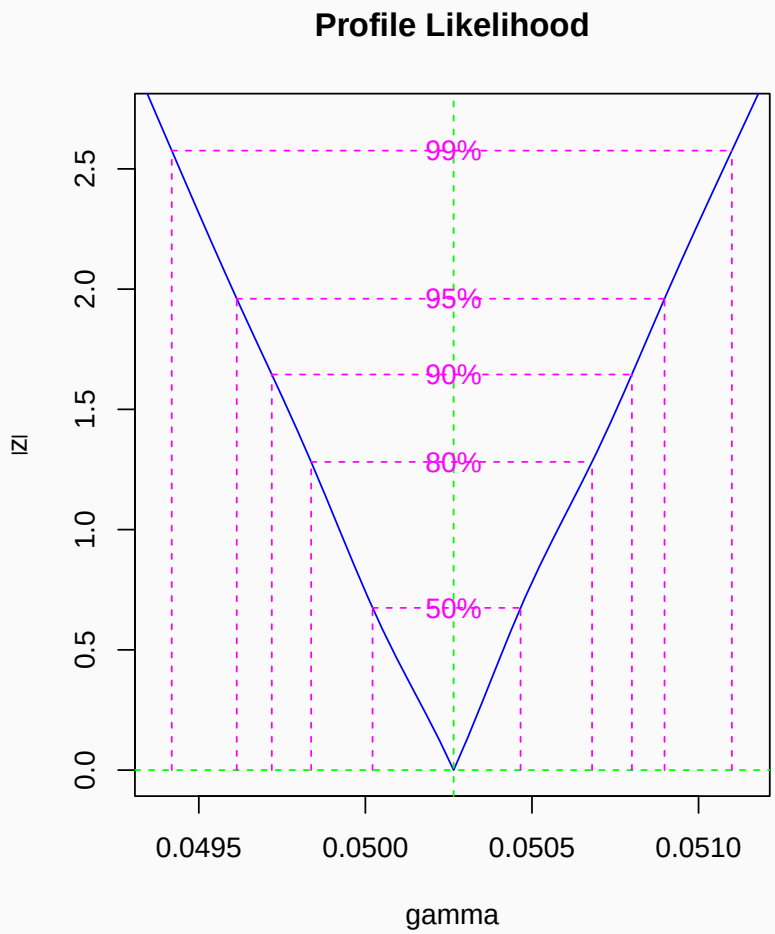
MLE results

```
Beta: 0.291 ± 0.003

Gamma: 0.05 ± 0

R0: 5.79
```

Uncertainty quantification with MLE

```r
# Profile likelihood for uncertainty
prof <- profile(fit_mle)
plot(prof, absVal = TRUE, main = "Profile Likelihood")
```





```r
# Confidence intervals
confint(fit_mle, level = 0.95)
```

```
            2.5 %     97.5 %
beta  0.28617565 0.29599639
gamma 0.04961407 0.05089814
```

MLE estimates vs true values

```
True Beta: 0.3

MLE Beta: 0.291

True Gamma: 0.1

MLE Gamma: 0.05
```

Model comparison with MLE

```r
# Fit different models and compare
# Model 1: SIR with Poisson
# Model 2: SIR with negative binomial

# Negative binomial likelihood
nll_nb <- function(beta, gamma, phi, data) {
  out <- ode(y = init_conds, times = data$time,
             func = sir_model, parms = c(beta = beta, gamma

  # Extract and scale predictions to cases
  predicted <- out[,"I"] * 1000

  # Negative Binomial negative log-likelihood
  nll <- -sum(dnbinom(data$observed, mu = predicted, size =

```

```r
# Fit NB model
fit_nb <- mle2(nll_nb,
               start = list(beta = 0.2, gamma = 0.1, phi =
               data = list(data = data),
               method = "L-BFGS-B",
               lower = c(0.01, 0.01, 0.1),
               upper = c(1.0, 0.5, 100))
```

Model comparison

```
Poisson AIC: 19650.9

Negative Binomial AIC: 3189.442

Delta AIC: -16461.45
```

Strengths of maximum likelihood

Statistical Rigor:

- Principled probabilistic framework
- Asymptotic optimality properties
- Natural uncertainty quantification
- Enables formal hypothesis testing

Practical benefits

- Confidence intervals and standard errors
- Model comparison via AIC/BIC
- Handles different error structures
- Extensible to complex models

Limitations of maximum likelihood

Computational Challenges:

- More complex than least squares
- Requires optimization algorithms
- Can get stuck in local minima
- Sensitive to starting values

Statistical assumptions

- Requires specification of error distribution
- Assumes model structure is correct
- Asymptotic properties may not hold
- Can be sensitive to outliers

Identifiability issues

- Still suffers from parameter identifiability
- Profile likelihood can be computationally expensive
- May not converge for complex models

R implementation practicals

- Let's turn to the tutorials

Comparison of methods

When to use each method

Use Least Squares When:

- Quick exploratory analysis needed
- Getting initial parameter estimates
- Computational speed is critical
- Simple error structure assumed

Use Maximum Likelihood When:

- Uncertainty quantification needed
- Comparing different models
- Formal statistical inference required
- Complex error structures present
- Publication-quality results needed

Advanced methods

Beyond least squares and MLE

Why We Need Advanced Methods:

- Parameter identifiability issues
- Complex error structures
- Model uncertainty
- Computational challenges
- Real-time fitting requirements

Advanced approaches

1. **Bayesian Methods** (MCMC)
2. **Particle Filtering**
3. **Approximate Bayesian Computation (ABC)**
4. **Ensemble Methods**

Bayesian methods: MCMC

Core Idea: Treat parameters as random variables with prior distributions

Bayes' Theorem:

$$P(\theta|\mathbf{y}) = \frac{P(\mathbf{y}|\theta)P(\theta)}{P(\mathbf{y})}$$

Advantages

- Natural uncertainty quantification
- Incorporates prior knowledge
- Handles parameter identifiability
- Model comparison via Bayes factors

Example with Stan

```
# Stan model for SIR fitting
"
// SIR model in Stan (walkthrough)
// Functions block: derivative of [S, I, R]
functions {
  vector SIR(real t, vector y, array[] real theta) {
    real S = y[1]; real I = y[2]; real R = y[3];
    real beta = theta[1]; real gamma = theta[2];
    vector[3] dydt;
    dydt[1] = -beta * S * I;
    dydt[2] =  beta * S * I - gamma * I;
    dydt[3] =  gamma * I;
    return dydt;
  }
}
```

Particle filtering

Core Idea: Sequential Monte Carlo method for state-space models

When to Use:

- Real-time parameter estimation
- State estimation in stochastic models
- Handling of missing data
- Time-varying parameters

Advantages

- Handles stochasticity naturally
- Real-time updates
- No assumption of constant parameters
- Robust to model misspecification

Example application

```
# Particle filter for SIR model
library(pomp)

# Define SIR model with stochasticity
sir_pomp <- pomp(
  data = data.frame(time = covid_times, cases = covid_obser
  times = "time",
  t0 = 0,
  rprocess = euler.sim(
    step.fun = "sir_step",
    delta.t = 0.1
  ),
  rmeasure = "cases_measure",
  dmeasure = "cases_dmeasure",
  initializer = "sir_init",
```

Approximate Bayesian computation (ABC)

Core Idea: Approximate posterior without likelihood evaluation

When to Use:

- Complex likelihoods
- Intractable models
- High-dimensional parameter spaces
- Model comparison

Algorithm:

1. Sample parameters from prior
2. Simulate data from model
3. Compare simulated to observed data
4. Accept if distance $<$ threshold

Advantages:

- No likelihood required
- Handles complex models
- Model comparison
- Intuitive approach

Ensemble methods

Core Idea: Combine multiple models or methods

Types:

- **Model Ensembles**: Average predictions from different models
- **Method Ensembles**: Combine LS, MLE, MCMC results
- **Bootstrap Ensembles**: Multiple fits with resampled data

Advantages:

- Reduces overfitting
- Quantifies model uncertainty
- More robust predictions
- Handles model selection uncertainty

Practical considerations

Common fitting problems

Convergence issues

- Poor starting values
- Flat likelihood surfaces
- Numerical instabilities
- Parameter bounds

Identifiability problems

- Multiple solutions
- Correlated parameters
- Insufficient data
- Model overparameterization

Solutions

- Multiple starting points
- Profile likelihood
- Data augmentation

Troubleshooting guide

If Optimization fails

1. Check starting values
2. Verify parameter bounds
3. Examine objective function
4. Try different algorithms
5. Simplify the model

If parameters are unidentifiable

1. Fix some parameters
2. Use additional data
3. Add regularization
4. Consider model reduction
5. Use prior information

If results are unrealistic

1. Check model assumptions
2. Verify data quality
3. Examine residuals
4. Test sensitivity
5. Consider alternative models

Best practices

Before fitting

- Understand your data
- Check model assumptions
- Set realistic parameter bounds
- Prepare multiple starting values

During fitting

- Monitor convergence
- Check for local minima
- Validate results
- Document everything

After fitting

- Assess goodness of fit
- Quantify uncertainty
- Test sensitivity
- Validate predictions

Software recommendations

R Packages

- `bbmle`: Maximum likelihood in R
- `rstan` and `cmdstanr`: Bayesian inference in R
- `pomp`: Particle filtering
- `brms`: Bayesian inference in R
- `nimble`: Bayesian inference in R

Specialized Software

- `Stan`: Probabilistic programming
- `OpenBUGS`: Bayesian inference
- `JAGS`: Bayesian analysis
- `PyMC`: Bayesian inference in Python
- `Turing.jl`: Bayesian inference in Julia

Conclusions

Key takeaways

Least Squares:

- Fast and intuitive
- Good for exploration
- Limited uncertainty quantification
- Sensitive to assumptions

Maximum Likelihood:

- Principled statistical framework
- Natural uncertainty quantification
- Enables model comparison
- More computationally intensive

Advanced Methods:

- Handle complex scenarios
- Provide robust uncertainty
- Require more expertise
- Often computationally expensive

Choosing the right method

For Quick Exploration: Least Squares

For Publication: Maximum Likelihood

For Complex Models: Bayesian Methods

For Real-time: Particle Filtering

For Model Comparison: ABC or MCMC

General Principle: Start simple, add complexity as needed

Final thoughts

Model fitting is both art and science:

- Requires domain expertise
- Demands statistical rigor
- Benefits from computational tools
- Needs careful validation

The goal is not just to fit models, but to:

- Understand disease dynamics
- Make reliable predictions
- Inform policy decisions
- Advance scientific knowledge

Any Questions?

Resources

Full courses (free)

- Model fitting and inference for infectious disease dynamics by Sebastian Funk, Anton Camacho, Helen Johnson, Amanda Minter, Kathleen O'Reilly and Nicholas Davies. Link

Core concepts

- Introduction to the Concept of Likelihood and Its Applications
- *Key considerations for model fitting and calibration*: Choices and trade-offs in inference with infectious disease models
- *A compilation of model fitting tutorials*: Tooling-up for infectious disease transmission modelling

Least squares

- *Key tutorial on the least squares method*: Fitting dynamic models to epidemic outbreaks with quantified uncertainty: A primer for parameter uncertainty, identifiability, and forecasts
- Fitting Epidemic Models to Data by James Holland Jones

MLE

- Fitting Epidemic Models to Data by James Holland Jones
- *R tutorial on MLE*: Estimating model parameters by maximum likelihood

MCMC

- Markov Chain Monte Carlo: an introduction for epidemiologists
- A simple introduction to Markov Chain Monte–Carlo sampling

pMCMC

- Introduction to particle Markov-chain Monte Carlo for disease dynamics modellers

ABC

- Approximate Bayesian Computation for infectious disease modelling

Others

- Bayesian workflow for disease transmission modeling in Stan
- POMP
- Odin and Monty

References

- Anderson, R.M. and May, R.M. (1991). *Infectious Diseases of Humans: Dynamics and Control*. Oxford University Press.
- Keeling, M.J. and Rohani, P. (2008). *Modeling Infectious Diseases in Humans and Animals*. Princeton University Press.
- Bolker, B. (2008). *Ecological Models and Data in R*. Princeton University Press.
- King, A.A. et al. (2016). "Statistical inference for partially observed Markov processes via the R package pomp." *Journal of Statistical Software*.
- Carpenter, B. et al. (2017). "Stan: A probabilistic programming language." *Journal of Statistical Software*.