Logistic Regression for Binary/Binomial Data

1. Purpose of the Model

The logistic regression model addresses problems where:

- The outcome is binary (success/failure) or binomial (count of successes)
- The probability of success depends on continuous covariates
- We want to model how the probability changes with the covariates

Example Problem: Challenger O-ring failures

- Outcome: Number of O-ring failures (0-6)
- Covariate: Temperature
- Goal: Predict failure probability at different temperatures

Key Advantages:

1. Handles non-linear probability relationships
2. Ensures probabilities stay in [0,1]
3. Provides interpretable coefficients
4. Can include multiple covariates

2. Prior and Likelihood Definition

Model Structure

For each observation i:

```
θ(x_i) = σ(α + βx_i)
y_i ~ Binomial(N_i, θ_i)
```

Where:

- `σ(z) = 1/(1 + exp(-z))` is the sigmoid function
- `α` is the intercept parameter
- `β` is the slope parameter
- `x_i` is the (standardized) covariate
- `N_i` is the number of trials

Prior Distribution

Independent Gaussian priors on parameters:

```
α ~ N(0, σ²_α)
β ~ N(0, σ²_β)
```

Likelihood

Binomial likelihood for each observation:

```
p(y_i|x_i, α, β) = Binomial(y_i|N_i, θ_i)
```

Full likelihood (assuming conditional independence):

```
p(y|x, α, β) = ∏_i Binomial(y_i|N_i, θ_i)
```

3. Posterior Distribution Computation

The posterior is not analytically tractable due to the non-conjugate model. We use **grid approximation**:

Grid Approximation Steps:

1. **Define grid**: Create discrete grid for (α, β)
2. **Evaluate**: Compute log $p(y, \alpha, \beta|x)$ at each grid point
3. **Normalize**: Convert to proper probability distribution

```
q(α_i, β_j) = p(y, α_i, β_j|x) / Z
Z = Σ_{i,j} p(y, α_i, β_j|x)
```

Alternative Methods (not shown):

- MCMC (Markov Chain Monte Carlo)
- Variational Inference
- Laplace Approximation

4. Posterior Predictive Distribution

For new temperature x*, the posterior predictive distribution:

```
p(y*|y, x, x*) = ∫∫ p(y*|x*, α, β)p(α, β|y, x)dαdβ
```

Using grid approximation:

```
p(y*|y, x, x*) ≈ Σ_{i,j} p(y*|x*, α_i, β_j)π_{ij}
```

Where $\pi_{ij} = q(\alpha_i, \beta_j)$ are the posterior probabilities.

5. Making Predictions with the Model

Point Estimates

1. **MLE**: $(\hat{\alpha}_{MLE}, \hat{\beta}_{MLE}) = \text{argmax } p(y|x, \alpha, \beta)$
2. **MAP**: $(\hat{\alpha}_{MAP}, \hat{\beta}_{MAP}) = \text{argmax } p(\alpha, \beta|y, x)$
3. **Posterior Mean**: $E[\alpha|y]$, $E[\beta|y]$

Plugin Approximations

For quick predictions, use point estimates:

```
θ̂(x) = σ(α̂ + β̂x)
```

Full Bayesian Predictions

For uncertainty quantification:

1. Sample (α, β) from posterior
2. Compute $\theta(x)$ for each sample
3. Calculate mean and credible intervals

Key Predictions:

- **Single failure probability**: $\theta(x)$
- **At least one failure**: $1 - (1 - \theta(x))^N$
- **Expected failures**: $N \times \theta(x)$
- **Credible intervals**: Quantiles of posterior samples

6. Model Interpretation

Coefficients:

- `α`: Log-odds when x = 0 (intercept)
- `β`: Change in log-odds per unit change in x
- `exp(β)`: Odds ratio per unit change in x

Effect of Temperature (Challenger example):

- Negative β means lower temperature → higher failure probability
- Magnitude of β indicates sensitivity to temperature
- Uncertainty in β reflects parameter uncertainty

7. Model Comparison

Aspect	Simple Beta-Binomial	Logistic Regression
Covariates	None	Yes
Parameter interpretation	Direct probability	Log-odds scale
Computational complexity	Low (conjugate)	Medium (grid/MCMC)
Flexibility	Limited	High
Use case	Homogeneous data	Heterogeneous data

8. Implementation Summary

python

```python
# 1. Setup model
model = LogisticRegression(x, y, N, sigma2_alpha=1.0, sigma2_beta=1.0)

# 2. Grid approximation
post_approx = GridApproximation2D(alphas, betas, model.log_joint)

# 3. Point estimates
alpha_MLE, beta_MLE = likelihood_grid.argmax
alpha_MAP, beta_MAP = posterior_grid.argmax

# 4. Posterior sampling
alpha_samples, beta_samples = post_approx.sample(key, n_samples)

# 5. Predictions
theta_samples = model.theta(x_new, alpha_samples, beta_samples)
mean_pred = theta_samples.mean(0)
ci_lower = jnp.quantile(theta_samples, 0.025, axis=0)
ci_upper = jnp.quantile(theta_samples, 0.975, axis=0)
```

9. Key Takeaways

1. **Non-conjugate model**: Requires approximation methods
2. **Temperature effect**: Clear relationship between temperature and failure
3. **Uncertainty matters**: Full Bayesian approach provides credible intervals
4. **Grid limitations**: Works for 2D but doesn't scale to many parameters
5. **Practical insight**: Cold temperatures dramatically increase failure risk

10. Extensions

- **Multiple covariates**: $f(x) = \alpha + \beta_1 x_1 + \beta_2 x_2 + \ldots$
- **Interaction terms**: $f(x) = \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2$
- **Non-linear effects**: Polynomial terms, splines
- **Hierarchical models**: Group-specific parameters
- **Different link functions**: Probit, complementary log-log