Gaussian Process Regression - Detailed Hand Calculations

1. Model Setup

Consider a simple 1D example with 3 training points:

- $X = [0, 1, 2]^T$
- $y = [0.5, 1.2, 0.8]^T$
- Test point: $x^* = 1.5$

Using squared exponential kernel with hyperparameters:

- $\kappa = 1.0$ (magnitude)
- $\ell = 1.0$ (lengthscale)
- $\sigma = 0.1$ (noise std dev)

2. Kernel Computations

2.1 Squared Exponential Kernel

```
k(x, x') = κ² exp(-½(x-x')²/ℓ²)
```

2.2 Training Covariance Matrix K(X,X)

```
K₁₁ = k(0,0) = 1² × exp(0) = 1.000
K₁₂ = k(0,1) = 1² × exp(-½×1²/1²) = exp(-0.5) = 0.607
K₁₃ = k(0,2) = 1² × exp(-½×4²/1²) = exp(-2) = 0.135
K₂₂ = k(1,1) = 1.000
K₂₃ = k(1,2) = exp(-0.5) = 0.607
K₃₃ = k(2,2) = 1.000

K = [1.000  0.607  0.135]
    [0.607  1.000  0.607]
    [0.135  0.607  1.000]
```

2.3 Adding Noise

```
C = K + σ²I = K + 0.01I

C = [1.010  0.607  0.135]
    [0.607  1.010  0.607]
    [0.135  0.607  1.010]
```

2.4 Test-Train Covariance K(x*, X)

```
k₁ = k(1.5, 0) = exp(-½×(1.5)²/1²) = exp(-1.125) = 0.325
k₂ = k(1.5, 1) = exp(-½×(0.5)²/1²) = exp(-0.125) = 0.882
k₃ = k(1.5, 2) = exp(-½×(0.5)²/1²) = exp(-0.125) = 0.882

k* = [0.325  0.882  0.882]
```

3. Posterior Mean Calculation

3.1 Solve $C^{-1}y = \alpha$

```
C × α = y
[1.010  0.607  0.135] [α₁]   [0.5]
[0.607  1.010  0.607] [α₂] = [1.2]
[0.135  0.607  1.010] [α₃]   [0.8]
```

Using matrix inversion (or linear system solver):

```
α ≈ [0.127]
    [0.874]
    [0.332]
```

3.2 Compute Posterior Mean

```
μ* = k*ᵀ × α
   = [0.325  0.882  0.882] × [0.127]
                             [0.874]
                             [0.332]
   = 0.325×0.127 + 0.882×0.874 + 0.882×0.332
   = 0.041 + 0.771 + 0.293
   = 1.105
```

4. Posterior Variance Calculation

4.1 Prior Variance at Test Point

```
k** = k(1.5, 1.5) = 1.000
```

4.2 Solve $C^{-1}k^* = v$

```
C × v = k*
```

Solution:

```
v ≈ [0.007]
    [0.744]
    [0.241]
```

4.3 Compute Posterior Variance

```
σ²* = k** - k*ᵀ × v
    = 1.000 - [0.325  0.882  0.882] × [0.007]
                                       [0.744]
                                       [0.241]
    = 1.000 - (0.002 + 0.656 + 0.213)
    = 1.000 - 0.871
    = 0.129
```

4.4 Standard Deviation

```
σ* = √0.129 = 0.359
```

5. Predictive Distribution

For observation y* at x* = 1.5:

```
p(y*|X,y) = N(1.105, 0.129 + 0.01)
          = N(1.105, 0.139)
```

95% confidence interval:

```
[μ* - 2σ*, μ* + 2σ*] = [1.105 - 2×0.373, 1.105 + 2×0.373]
                     = [0.359, 1.851]
```

6. Marginal Likelihood

6.1 Cholesky Decomposition

```
C = LLᵀ where L is lower triangular
```

6.2 Log Marginal Likelihood

```
log p(y|X,θ) = -½yᵀC⁻¹y - ½log|C| - ½n log(2π)
```

Components:

1. Data fit term: $-\frac{1}{2}y^T C^{-1} y$
2. Complexity penalty: $-\frac{1}{2}\log|C| = -\Sigma\log(L_{ii})$
3. Normalization: $-\frac{1}{2}n \log(2\pi)$

7. Kernel Properties

Squared Exponential Properties:

- Infinitely differentiable (very smooth)
- Universal approximator
- Stationary and isotropic
- Characterized by lengthscale ℓ

Effect of Hyperparameters:

- **κ (magnitude)**: Controls function variance
  - Large κ → large function variations
  - Small κ → small function variations
- **ℓ (lengthscale)**: Controls smoothness
  - Large ℓ → smooth, slowly varying functions
  - Small ℓ → rough, quickly varying functions
- **σ (noise)**: Controls uncertainty
  - Large σ → high observation noise
  - Small σ → low observation noise

8. Computational Complexity

- Training: $O(n^3)$ due to matrix inversion
- Prediction mean: $O(n^2)$ per test point
- Prediction variance: $O(n^2)$ per test point
- Storage: $O(n^2)$ for kernel matrix

9. Practical Considerations

1. **Numerical Stability**:
   - Add jitter (small diagonal term) to K
   - Use Cholesky decomposition
   - Avoid explicit matrix inversion
2. **Hyperparameter Optimization**:
   - Maximize log marginal likelihood
   - Use gradient-based optimization
   - Consider multiple random restarts
3. **Scalability**:
   - For large datasets, use sparse GPs
   - Inducing points approximation
   - Local approximations