Principal Component Analysis vs Factor Analysis

MA2003B - Application of Multivariate Methods in Data Science

MA2003B Course Team

Tecnológico de Monterrey

2025-10-03

Introduction to Principal Component Analysis and Factor Analysis

Why Dimensionality Reduction?

Dimensionality reduction is a fundamental technique in multivariate analysis that seeks to:

- Handle high-dimensional data effectively
- Reduce computational complexity
- Remove noise and redundancy
- Improve model interpretability
- Enable data visualization

Two Main Approaches

- **Principal Component Analysis (PCA)**: Data-driven approach, variance maximization
- **Factor Analysis (FA)**: Model-based approach, identification of latent constructs

Principal Component Analysis (PCA)

What is PCA?

PCA is a dimensionality reduction technique that transforms correlated variables into uncorrelated principal components, maximizing the variance along each new axis and ordering the components by explained variance.

Mathematical Foundation

Given a data matrix $\boldsymbol{X}$ with n observations and p variables:

1. Center the data: $\boldsymbol{X}_{\text{centered}} = \boldsymbol{X} - \overline{\boldsymbol{X}}$
2. Calculate the covariance matrix: $\boldsymbol{S} = \left(\frac{1}{n-1}\right)\boldsymbol{X}_{\text{centered}}^T \boldsymbol{X}_{\text{centered}}$
3. Find eigenvalues λ_i and eigenvectors $\boldsymbol{v}_i$ of $\boldsymbol{S}$
4. Principal components: $\mathbf{PC}_i = \boldsymbol{X}_{\text{centered}} \boldsymbol{v}_i$

PCA Example

```python
import numpy as np
from sklearn.decomposition import PCA

# Simple 3x2 data matrix
X = np.array([[5, 3],
              [3, 1],
              [1, 3]])

# Apply PCA
pca = PCA()
X_transformed = pca.fit_transform(X)

# Results
```

```python
eigenvalues = pca.explained_variance_
variance_ratio = pca.explained_variance_ratio_

print(f"Eigenvalues: {eigenvalues}")
print(f"PC1 explains {variance_ratio[0]:.1%} of variance")
```

Key Results

- PC1 explains the most variance (highest eigenvalue)
- Components are uncorrelated by construction
- Original data can be reconstructed from the components

Component Retention Criteria

- **Kaiser Criterion**: Keep components with $\lambda_i > 1$
- **Scree Plot**: Look for the "elbow" in the eigenvalue plot
- **Cumulative Variance**: Retain enough components for the desired variance (e.g., 80%)
- **Parallel Analysis**: Compare with eigenvalues of random data

Factor Analysis

What is Factor Analysis?

Factor analysis assumes that observed variables are linear combinations of common factors (shared latent constructs) and unique factors (variable-specific variance).

The Common Factor Model

For each observed variable x_j:

$$x_j = \mu_j + \sum_{i=1}^{m} \lambda_{ji} f_i + \varepsilon_j$$

Where:

- λ_{ji}: Factor loading (correlation between x_j and f_i)
- f_i: Common factor (latent variable)
- ε_j: Unique factor (error term)

Factor Analysis Example

```python
import numpy as np
from factor_analyzer import FactorAnalyzer

# Correlation matrix
R = np.array([[1.00, 0.60, 0.48],
              [0.60, 1.00, 0.72],
              [0.48, 0.72, 1.00]])

# Perform Factor Analysis
fa = FactorAnalyzer(n_factors=1, rotation=None)
fa.fit(R)
```

```python
# Results
loadings = fa.loadings_
communalities = fa.get_communalities()
uniqueness = fa.get_uniquenesses()

print(f"Factor loadings:\\n{loadings}")
print(f"Communalities: {communalities}")
print(f"Uniquenesses: {uniqueness}")
```

Key Concepts

- **Loadings**: Correlations between variables and factors
- **Communalities**: Variance explained by common factors
- **Uniquenesses**: Variable-specific variance

Factor Rotation

Factor rotation is a technique applied after factor extraction to make factors easier to interpret. It "rotates" the factor axes to achieve **simple structure**: each variable should load highly on only one factor and low on others.

The problem: Initial extraction often produces factors where many variables have moderate loadings on multiple factors, making interpretation difficult.

The solution: Rotation redistributes variance among factors without changing the total variance explained.

Two main types:

- **Orthogonal rotation** (e.g., Varimax): Keeps factors uncorrelated (independent)
  - Simpler interpretation, but may be unrealistic if constructs actually correlate
- **Oblique rotation** (e.g., Promax, Oblimin): Allows factors to correlate
  - More realistic for real-world data, slightly more complex interpretation

Method	Description	Assumption	Use Case
Varimax	Orthogonal rotation	Uncorrelated factors	Simple structure
Promax	Oblique rotation	Factors may correlate	Realistic models
Quartimax	Simplify variables	Balance factors	General use
Oblimin	Oblique rotation	Flexible correlation	Complex data

Why Rotate?

- Improve the interpretability of factor loadings
- Achieve "simple structure" (variables that load highly on few factors)
- Different rotations can reveal different substantive interpretations

Comparison: PCA vs Factor Analysis

Key Differences

Aspect	PCA	Factor Analysis
Objective	Variance maximization	Latent construct identification
Model	Data-based	Theory-based
Components/Factors	All variance	Only common variance
Rotation	Not typically used	Essential for interpretation
Assumptions	Minimal	Multivariate normality
Estimation	Eigenvalue decomposition	Maximum likelihood/PAF
Output	Principal components	Factor loadings

When to Use Each Method

Use PCA when:

- Data reduction is the main goal
- No theoretical model exists
- All variance is of interest
- Prediction is the objective
- Data visualization is needed

Use Factor Analysis when:

- Identifying latent constructs
- The analysis is theory-driven
- Developing a measurement model
- Understanding relationships between variables
- Performing scale validation/development

Complete Analysis Workflow

Before performing Factor Analysis, we must verify that our data is suitable:

Kaiser-Meyer-Olkin (KMO) Test

- Measures if your variables have enough in common to group into factors
- High KMO means variables cluster well together; low KMO means they are too independent
- Ranges from 0 to 1
- Interpretation: > 0.9 excellent, > 0.8 good, > 0.6 acceptable, < 0.5 unacceptable
- **What it tests:** Are correlations strong enough for FA to be useful?

Bartlett's Test of Sphericity

- Tests if the correlation matrix is significantly different from an identity matrix
- Uses chi-square statistic
- Interpretation: $p < 0.05$ means variables are correlated (good for FA)

- **What it tests:** Are variables correlated at all?

Key difference: Bartlett tests if correlations exist, KMO tests if they are strong enough.

```
import numpy as np
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from factor_analyzer import FactorAnalyzer
from factor_analyzer.factor_analyzer import calculate_kmo,
calculate_bartlett_sphericity

# 1. Load and prepare data
X = np.random.randn(100, 5)  # Your data here

# 2. Standardize
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# 3. Check suitability for FA
kmo_all, kmo_model = calculate_kmo(X_scaled)
chi_square_value, p_value = calculate_bartlett_sphericity(X_scaled)

print(f"KMO: {kmo_model:.3f} (>0.6 is good)")
print(f"Bartlett's test p-value: {p_value:.3f} (<0.05 is good)")

# 4. Determine number of factors
pca = PCA()
pca.fit(X_scaled)
eigenvalues = pca.explained_variance_
n_factors = sum(eigenvalues > 1)  # Kaiser criterion

# 5. Perform Factor Analysis
fa = FactorAnalyzer(n_factors=n_factors, rotation='varimax')
fa.fit(X_scaled)

# 6. Compare results
loadings = fa.loadings_
communalities = fa.get_communalities()
variance_explained = fa.get_factor_variance()
```

Applications and Best Practices

Real-World Applications

Finance

- Risk factors in stock markets
- Portfolio optimization
- Credit scoring models

Health

- Patient satisfaction surveys
- Symptom clustering
- Quality of life measures

Marketing

- Customer segmentation
- Brand perception studies
- Product attribute analysis

Social Sciences

- Personality assessment
- Attitude measurement
- Educational testing

Best Practices

Data Preparation

- Ensure adequate sample size (5-10 observations per variable)
- Check for multivariate normality
- Handle missing data appropriately
- Consider variable standardization

Model Selection

- Use KMO and Bartlett's tests for FA suitability
- Compare multiple factor retention criteria
- Consider both orthogonal and oblique rotations
- Validate results with cross-validation

Interpretation

- Focus on the substantive meaning of the factors
- Use factor loadings > 0.3 for interpretation
- Consider correlations between factors in oblique rotations
- Validate with external criteria when possible

Conclusion

Key Takeaways

- **PCA** and **Factor Analysis** serve different but complementary purposes
- Choose the method based on research objectives and data characteristics
- Always validate assumptions and interpret results substantively
- Modern software makes implementation straightforward

Next Steps

- Practice with real datasets
- Compare PCA and FA on the same data
- Explore advanced techniques (confirmatory FA, structural equation modeling)
- Apply to your own research questions

References

Key References

- **Fabrigar, L. R., & Wegener, D. T.** (2011). Exploratory Factor Analysis. Oxford University Press.
- **Hair, J. F., et al.** (2019). Multivariate Data Analysis. Cengage Learning.
- **Jolliffe, I. T.** (2002). Principal Component Analysis. Springer.
- **Tabachnick, B. G., & Fidell, L. S.** (2013). Using Multivariate Statistics. Pearson.

Software Resources

- Python: `scikit-learn`, `factor-analyzer`, `statsmodels`
- R: `psych`, `FactoMineR`, `lavaan`
- SPSS: Factor Analysis Module
- SAS: PROC FACTOR

Online Resources

- UCLA Statistical Consulting Group
- StatQuest YouTube channel
- Towards Data Science articles