Principal Component Analysis

Dr. Juliho Castillo

Tecnológico de Monterrey

2025-09-29

Table of contents

Índice

Part I: PCA Theory

Principal Component Analysis Theory

Understanding PCA: The Big Picture

Imagine you're a photographer trying to capture the best view of a 3D sculpture.

You want to find the **single best angle** that shows the most interesting features and variations of the sculpture. That's essentially what Principal Component Analysis does with data!

What PCA does in simple terms:

- Takes your data with many variables (dimensions)
- Finds the «best directions» to look at your data

- These directions capture the most variation and patterns
- Reduces complexity while keeping the most important information

Why is this useful?

- Makes complex data easier to visualize and understand
- Removes noise and redundant information
- Helps identify the most important patterns in your data

Refresher: What is PCA?

- Principal Component Analysis (PCA) is a linear method for **dimension reduction**.
- It finds orthogonal directions (principal components) that capture the largest possible variance in the data.
- PCA produces new variables (components) that are linear combinations of the original observed variables.
- Use cases: visualization, noise reduction, pre-processing before supervised learning, and exploratory data analysis.

Refresher: Eigen Decomposition

Eigen decomposition is a fundamental matrix factorization technique used in multivariate analysis.

Definition: For a square matrix $\boldsymbol{A}$, if it can be diagonalized, we can write: $\boldsymbol{A} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^{-1}$

Where:

- $\boldsymbol{D}$ is a diagonal matrix containing the eigenvalues $\lambda_1, \lambda_2, ..., \lambda_n$
- $\boldsymbol{P}$ is the matrix whose columns are the eigenvectors $\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_n$
- Each eigenvector satisfies: $\boldsymbol{A}\boldsymbol{v}_j = \lambda_j \boldsymbol{v}_j$

Deeper Meaning and Significance:

- **Eigenvalues** (λ_j) measure the «strength» or «importance» of each underlying pattern in your data
- **Eigenvectors** (v_j) reveal the «direction» or «profile» of these patterns
- This decomposition is fundamental because it **separates complex multivariate relationships into independent, interpretable components**
- In factor analysis context: eigenvalues help determine how many meaningful factors exist, while eigenvectors show how variables cluster together

For symmetric matrices (like covariance matrices in PCA/FA):

- The eigenvectors are orthonormal ($\boldsymbol{P}^\top \boldsymbol{P} = \boldsymbol{I}$)
- The decomposition simplifies to: $\boldsymbol{A} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^\top$

Geometric interpretation: Eigenvectors represent directions of maximum variance, eigenvalues represent the magnitude of variance in those directions.

In multivariate statistics: This decomposition underlies both PCA (principal components) and Factor Analysis (latent factors).

For detailed matrix algebra foundations, see Appendix.

Why PCA Works: The Big Idea

The Goal: Find the direction that captures the most variance in your data

The Problem: We want to maximize variance, but prevent the solution from becoming infinite

The Solution in Simple Terms:

1. **What we want:** Direction with maximum variance
2. **Constraint:** Direction must have unit length (prevents infinity)
3. **Mathematical magic:** This leads to the eigenvalue problem

4. **Key insight:** $Sv = \lambda v$
5. **Result:** Largest eigenvalue = maximum variance

Why This Works:

- Eigenvalues tell us how much variance each direction captures
- Eigenvectors tell us what those directions are
- We pick the directions with the most variance

In Practice:

- Computer finds eigenvalues and eigenvectors
- We sort them from largest to smallest
- First few capture most of the interesting patterns

Mathematical Formulation: Foundation

Let $\boldsymbol{x} \in \mathbb{R}^p$ be a random vector with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.

Data Matrix Representation:

- Data matrix $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ with n observations and p variables
- Each row $\boldsymbol{x}_i^\top$ is an observation vector
- Centered data: $\boldsymbol{X}_c = \boldsymbol{X} - \mathbf{1}_n \overline{\boldsymbol{x}}^\top$ where $\mathbf{1}_n$ is vector of ones

Sample Covariance Matrix:

$$\boldsymbol{S} = \frac{1}{n-1}\boldsymbol{X}_c^\top \boldsymbol{X}_c \tag{1}$$

Pointwise Form:

$$s_{ij} = \frac{1}{n-1}\sum_{l=1}^{n}(x_{li} - \overline{x}_i)\big(x_{lj} - \overline{x}_j\big) \tag{2}$$

Eigenvalue Problem: Find eigenvalues $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_p \geq 0$ and orthonormal eigenvectors $\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_p$ such that:

$$\boldsymbol{S}\boldsymbol{v}_j = \lambda_j \boldsymbol{v}_j \quad \text{for } j = 1, 2, ..., p \tag{3}$$

Mathematical Formulation: Spectral Decomposition

Spectral Decomposition of Covariance Matrix:

$$\boldsymbol{S} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^\top = \sum_{j=1}^{p} \lambda_j \boldsymbol{v}_j \boldsymbol{v}_j^\top \tag{4}$$

Pointwise Form:

$$s_{ij} = \sum_{l=1}^{p} \lambda_l v_{il} v_{jl} \tag{5}$$

where:

- $\boldsymbol{V} = \left[\boldsymbol{v}_1 \mid \boldsymbol{v}_2 \mid ... \mid \boldsymbol{v}_p\right] \in \mathbb{R}^{p \times p}$ (eigenvector matrix)
- $\boldsymbol{\Lambda} = \operatorname{diag}\left(\lambda_1, \lambda_2, ..., \lambda_p\right)$ (diagonal eigenvalue matrix)
- $\boldsymbol{V}^\top \boldsymbol{V} = \boldsymbol{V}\boldsymbol{V}^\top = \boldsymbol{I}_p$ (orthonormality condition)

Principal Components: The j-th principal component for observation i is:

$$z_{ij} = \boldsymbol{v}_j^\top (\boldsymbol{x}_i - \overline{\boldsymbol{x}}) = \boldsymbol{v}_j^\top \boldsymbol{x}_{ci} \tag{6}$$

Pointwise Form:

$$z_{ij} = \sum_{l=1}^{p} v_{jl}(x_{il} - \overline{x}_l) \tag{7}$$

Component Score Matrix:

$$\boldsymbol{Z} = \boldsymbol{X}_c\boldsymbol{V} \in \mathbb{R}^{n\times p} \tag{8}$$

Mathematical Formulation: Variance Properties

Variance of Principal Components:

$$\mathrm{Var}\left(Z_j\right) = \mathrm{Var}\left(\boldsymbol{v}_j^\top \boldsymbol{X}_c\right) = \boldsymbol{v}_j^\top \boldsymbol{S} \boldsymbol{v}_j = \lambda_j \tag{9}$$

Total Variance Decomposition:

$$\mathrm{tr}(\boldsymbol{S}) = \sum_{j=1}^{p} s_{jj} = \sum_{j=1}^{p} \lambda_j \tag{10}$$

Proportion of Variance Explained: By component j: $\rho_j = \frac{\lambda_j}{\sum_{k=1}^{p} \lambda_k}$

Cumulative: $\rho_{1:k} = \frac{\sum_{j=1}^{k} \lambda_j}{\sum_{j=1}^{p} \lambda_j}$

Reconstruction Formula: Using first k components: $\hat{\boldsymbol{x}} = \overline{\boldsymbol{x}} + \sum_{j=1}^{k} z_j \boldsymbol{v}_j$

Pointwise Form: $\hat{x}_i = \overline{x}_i + \sum_{j=1}^{k} z_j v_{ji}$

Mean Squared Reconstruction Error: $\mathrm{MSE} = \frac{1}{n} \|\boldsymbol{X}_c - \boldsymbol{Z}_{1:k} \boldsymbol{V}_{1:k}^\top\|_F^2 = \sum_{j=k+1}^{p} \lambda_j$

From Concept to Computation: PCA Algorithm

Ready to turn theory into practice?

The PCA algorithm is like a recipe for finding the best viewpoints of your data. Think of it as teaching a computer to be that photographer we mentioned earlier!

What the algorithm does step-by-step:

1. **Preparation**: Clean and standardize the data (like focusing the camera)
2. **Find relationships**: Calculate how variables relate to each other
3. **Discover directions**: Find the best angles (principal components)
4. **Transform data**: Project data onto these new viewpoints
5. **Decide**: How many viewpoints do we actually need?

Why follow this exact sequence?

- Each step builds on the previous one
- Mathematical guarantees that we find the **optimal** solution
- Practical choices (like standardization) can dramatically affect results

Let’s see the detailed mathematical recipe:

Algorithm: Principal Component Analysis

Input: Data matrix $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ (n observations, p variables), standardization choice **Output:** Principal components $\boldsymbol{V}$, eigenvalues $\boldsymbol{\Lambda}$, component scores $\boldsymbol{Z}$

1. **Prepare Your Data**
   - **Different units?** (age vs income) → Standardize all variables
   - **Same units?** (all test scores) → Just center the data
   - **Rule of thumb:** When in doubt, standardize
2. **Calculate Relationships Between Variables**

- Compute correlation matrix: How do variables relate?
- This captures all the patterns in your data

3. **Find the Best Directions** (Eigenvalues & Eigenvectors)
    - Computer finds the directions with most variance
    - Eigenvalues = how much variance each direction captures
    - Eigenvectors = what those directions are

4. **Transform Your Data**
    - Project data onto the new directions
    - Get principal component scores for each observation

5. **Decide How Many Components to Keep**
    - Use Kaiser rule: keep eigenvalues > 1

- Or pick enough to explain 70-80% of variance
- Fewer components = simpler interpretation

PCA Algorithm: Simple Numerical Example

Given Data: 3 observations, 2 variables

$$\boldsymbol{X} = \begin{pmatrix} 5 & 3 \\ 3 & 1 \\ 1 & 3 \end{pmatrix} \quad \text{and} \quad \overline{\boldsymbol{x}} = \begin{pmatrix} 3 \\ 2.33 \end{pmatrix} \tag{11}$$

Step 1: Center the data

$$\boldsymbol{X}_c = \boldsymbol{X} - \mathbf{1}_3 \overline{\boldsymbol{x}}^\top = \begin{pmatrix} 5 & 3 \\ 3 & 1 \\ 1 & 3 \end{pmatrix} - \begin{pmatrix} 3 & 2.33 \\ 3 & 2.33 \\ 3 & 2.33 \end{pmatrix} = \begin{pmatrix} 2 & 0.67 \\ 0 & -1.33 \\ -2 & 0.67 \end{pmatrix} \tag{12}$$

Step 2: Compute sample covariance matrix

$$\boldsymbol{S} = \frac{1}{2} \boldsymbol{X}_c^\top \boldsymbol{X}_c = \frac{1}{2} \begin{pmatrix} 2 & 0 & -2 \\ 0.67 & -1.33 & 0.67 \end{pmatrix} \begin{pmatrix} 2 & 0.67 \\ 0 & -1.33 \\ -2 & 0.67 \end{pmatrix} = \begin{pmatrix} 4 & -0.67 \\ -0.67 & 1.33 \end{pmatrix} \tag{13}$$

Step 3: Solve eigenvalue problem $\boldsymbol{S}\boldsymbol{v} = \lambda \boldsymbol{v}$

- Characteristic equation: $\det(\boldsymbol{S} - \lambda \boldsymbol{I}) = (4 - \lambda)(1.33 - \lambda) - 0.67^2 = 0$
- Eigenvalues: $\lambda_1 = 4.45$, $\lambda_2 = 0.88$

- Eigenvectors: $\boldsymbol{v}_1 = \binom{0.95}{-0.32}$, $\boldsymbol{v}_2 = \binom{0.32}{0.95}$

Step 4: Compute PC scores

$$\boldsymbol{Z} = \boldsymbol{X}_c \boldsymbol{V} = \begin{pmatrix} 2 & 0.67 \\ 0 & -1.33 \\ -2 & 0.67 \end{pmatrix} \begin{pmatrix} 0.95 & 0.32 \\ -0.32 & 0.95 \end{pmatrix} = \begin{pmatrix} 1.69 & 1.28 \\ 0.43 & -1.26 \\ -2.12 & 0.00 \end{pmatrix} \tag{14}$$

Interpretation: PC1 explains $\frac{4.45}{5.33} = 83.5\%$ of total variance

Python Implementation: PCA Example

```
import numpy as np
from sklearn.decomposition import PCA
import pandas as pd

# Step 1: Create the data
X = np.array([[5, 3],
              [3, 1],
              [1, 3]])

# Step 2: Apply PCA
```

```python
pca = PCA()
X_transformed = pca.fit_transform(X)

# Step 3: Get results
eigenvalues = pca.explained_variance_
eigenvectors = pca.components_.T
variance_ratio = pca.explained_variance_ratio_

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

Output matches our manual calculation!

Deciding how many components to retain

Common heuristics and formal approaches:

- Kaiser criterion: keep components with eigenvalue > 1 (applies when using correlation matrix).
- Cumulative variance: keep the smallest number of components that explain a target (e.g., 70–90%) of total variance.
- Scree plot: look for the «elbow» where additional components contribute little incremental variance.

- Parallel analysis: compare empirical eigenvalues to those obtained from random data — keep components with larger eigenvalues than random.
  - **How it works**: Generate random datasets with same dimensions (n observations × p variables) as your data
  - **Compare**: For each component k, if $\lambda_k \text{ (actual)} > \lambda_k \text{ (random)}$, retain component k
  - **Advantage**: Accounts for sampling error and prevents over--extraction
  - **Conservative approach**: Often retains fewer components than Kaiser criterion

Algorithm: Component/Factor Retention Decision

Input: Eigenvalues $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_p$, variance threshold α

Output: Optimal number of components/factors k^*

1. **Kaiser Criterion** (Rule of Thumb)
   - Keep components with eigenvalue > 1
   - **Why?** Each component should explain more variance than a single variable

- **Easy rule:** Count how many eigenvalues are bigger than 1

2. **Cumulative Variance** (Practical Goal)
   - Keep enough components to explain 70-80% of total variance
     - **Example:** If first 3 components explain 75%, keep 3
       - **Trade-off:** More components = more complexity

3. **Scree Plot** (Visual Method)
   - Plot eigenvalues from largest to smallest
     - Look for the «elbow» - where the line flattens out
       - Keep components before the elbow

4. **Parallel Analysis** (Statistical Test)
   - Compare your eigenvalues to random data
     - Keep components larger than random ones
       - **Software does this automatically**

5. **Final Decision**
   - Use multiple methods and find agreement

- When in doubt, choose fewer components (simpler is better)
  - **Recommendation:** Use parallel analysis as primary criterion

Component Retention: Simple Numerical Example

Given Eigenvalues: From 5-variable correlation matrix

$$\lambda = [2.8, 1.2, 0.7, 0.2, 0.1] \tag{15}$$

Step 1: Kaiser Criterion

$$k_{\text{Kaiser}} = |\{j : \lambda_j > 1\}| = |\{1, 2\}| = 2 \tag{16}$$

components

Step 2: Cumulative Variance (80% threshold)

- Total variance: $\sum \lambda_j = 5.0$
- Cumulative proportions: $[0.56, 0.80, 0.94, 0.98, 1.00]$
- $k_{\text{variance}} = \min\{j : \rho_j \geq 0.80\} = 2$ components

Step 3: Parallel Analysis (simplified)

- Random eigenvalues (average): $\overline{\lambda}^{\text{random}} = [1.4, 1.1, 0.9, 0.7, 0.5]$
- Compare: $\lambda_j > \overline{\lambda}_j^{\text{random}}$
  - Factor 1: $2.8 > 1.4$ ✓
  - Factor 2: $1.2 > 1.1$ ✓
  - Factor 3: $0.7 < 0.9$ ×
- $k_{\text{parallel}} = 2$ components

Step 4: Consensus Decision

$$k^* = \text{consensus}(2, 2, 2) = 2 \tag{17}$$

components

Result: All criteria agree → retain 2 components explaining 80% of variance

Python Implementation: Component Retention

```python
import numpy as np
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

# Simulated data with 5 variables
np.random.seed(42)
X = np.random.randn(100, 5)

# Apply PCA
```

```python
pca = PCA()
pca.fit(X)

eigenvalues = pca.explained_variance_
cumvar = pca.explained_variance_ratio_.cumsum()

# Kaiser criterion
n_kaiser = sum(eigenvalues > 1)

# Cumulative variance (80% threshold)
n_cumvar = np.argmax(cumvar >= 0.8) + 1

print(f"Kaiser criterion: {n_kaiser} components")
print(f"80% variance: {n_cumvar} components")
print(f"Cumulative variance: {cumvar}")
```

```python
# Scree plot
plt.plot(range(1, 6), eigenvalues, 'bo-')
plt.axhline(y=1, color='r', linestyle='--')
plt.title('Scree Plot')
plt.show()
```

Algorithm: PCA Data Analysis Checklist

Input: Raw data matrix, research objectives

Output: Validated PCA results and interpretation

1. **Data Quality Assessment**
   - Check for missing values; handle via imputation or deletion
   - Detect outliers using Mahalanobis distance or visualization
   - **if** outliers are excessive **then** consider robust PCA methods
2. **Variable Scaling Decision**
   - Examine variable scales and units

- **if** variables have different scales **then**
  - Standardize: Use correlation matrix for PCA
- **else**
  - Use covariance matrix for PCA

3. **Component Interpretation**
   - Examine loading matrix $\boldsymbol{V}$
   - **for** each component j **do**
     - Identify variables with $|v_{ij}| > 0.3$ (substantial loading)
     - Name component based on dominant variables

4. **Results Validation and Reporting**
   - Generate eigenvalue table with variance proportions
   - Create scree plot for visual component selection

- Report cumulative variance explained
- Include rotated component matrix if rotation applied

Part II: PCA Examples

Example 1A: Educational Assessment PCA

Educational Assessment: PCA Analysis

This section demonstrates PCA using controlled synthetic data with known factor structure to validate the method and teach key concepts.

- **Dataset**: Student assessment data with 6 variables (100 students)
- **Research Question**: Can PCA recover the underlying ability factors? How does it separate meaningful structure from noise?
- **Method**: Standardized PCA on synthetic data with known latent factors
- **Scripts**: `educational_pca.py`

Dataset: Student Assessment Variables

Six variables representing different aspects of student ability:

- **MathTest**: Mathematics assessment score
- **VerbalTest**: Verbal reasoning assessment score
- **SocialSkills**: Social competency rating
- **Leadership**: Leadership ability rating
- **RandomVar1, RandomVar2**: Pure noise controls

Known Factor Structure (Ground Truth)

Ground Truth for Validation:

- *Intelligence Factor*: Affects MathTest (0.85 loading) and VerbalTest (0.80 loading)
- *Personality Factor*: Affects SocialSkills (0.85 loading) and Leadership (0.80 loading)
- Measurement error added to all meaningful variables (0.2-0.25 noise levels)

PCA Results: Factor Recovery

Running `educational_pca.py` reveals clear factor structure:

Component	Eigenvalue	% Variance	Cumulative %
PC1	2.203	36.7%	36.7%
PC2	1.608	26.8%	63.5%
PC3	0.842	14.0%	77.6%
PC4	0.736	12.3%	89.8%
PC5	0.322	5.4%	95.2%

PC6	0.289	4.8%	100.0%

- **Kaiser Criterion**: Retain PC1-PC2 (eigenvalues > 1.0)
- **Scree Test**: Clear elbow after PC2
- **Variance**: Two factors explain 63.5% of total variance

Component Loadings: Structure Discovery

Loadings matrix reveals underlying factor structure:

Variable	PC1	PC2	PC3
MathTest	0.489	0.502	−0.148
VerbalTest	0.467	0.518	0.184
SocialSkills	0.488	−0.483	0.345
Leadership	0.466	−0.498	−0.412
RandomVar1	0.325	0.124	0.634
RandomVar2	−0.283	−0.032	0.502

- **PC1**: General ability factor (all meaningful variables 0.47-0.49)
- **PC2**: Cognitive vs. Social separation (positive: Math/Verbal, negative: Social/Leadership)
- **Noise Validation**: Random variables show weaker, inconsistent patterns

PCA Interpretation: Method Validation

Factor Recovery Validation (comparing to ground truth):

- *Structure Detection*: PCA successfully identifies 2-factor structure
- *Meaningful vs. Noise*: Max loading for random variables (0.325) < meaningful variables (0.47)
- *Factor Separation*: PC2 cleanly separates cognitive (Math/Verbal) from social (Social/Leadership) abilities

Practical Insights:

- PC1 captures general «ability» factor common in educational assessments
- PC2 reveals specific cognitive vs. social skill dimensions
- Noise components (PC5-PC6) have eigenvalues < 0.35, clearly distinguishable

Example 2A: European Stock Markets PCA

European Stock Markets: PCA Analysis

This section demonstrates PCA applied to financial markets using synthetic European stock market data.

- **Dataset**: 4 major European indices (DAX, SMI, CAC, FTSE) over 1,860 trading days
- **Research Question**: How integrated are European financial markets? Can we identify common market factors?
- **Method**: Standardized PCA on correlation matrix of daily returns
- **Scripts**: `invest_pca.py`

Dataset: European Market Indices

- **DAX (Germany)**: Frankfurt Stock Exchange — largest European economy
- **SMI (Switzerland)**: Swiss Market Index — major financial center
- **CAC (France)**: Paris Stock Exchange — core eurozone market
- **FTSE (UK)**: London Stock Exchange — major international hub

PCA Results: Market Integration

Running `invest_pca.py` reveals extraordinary market integration:

Component	Eigenvalue	% Variance	Cumulative %
PC1	3.895	97.3%	97.3%
PC2	0.092	2.3%	99.6%
PC3	0.011	0.3%	99.9%
PC4	0.004	0.1%	100.0%

- **Dominant PC1**: Captures almost all variance (97.3%)

- **Kaiser Criterion**: Only PC1 has eigenvalue > 1.0
- **Interpretation**: European markets move as a single integrated system
- **Implication**: Extremely limited diversification within Europe

Component Loadings: Perfect Market Synchronization

All European markets load equally on PC1 (common market factor):

Market Index	PC1 Loading	PC2 Loading
DAX (Germany)	0.501	0.502
SMI (Switzerland)	0.501	−0.513
CAC (France)	0.500	0.351

FTSE (UK)	0.499	−0.625

- **PC1 Interpretation**: Uniform loadings (0.50) = perfect market integration
- **PC2 Interpretation**: Subtle Brexit effect (FTSE vs continental markets)
- **Financial Reality**: Global/EU-wide factors dominate individual market performance

Financial Interpretation: Systematic Risk Dominance

PC1 as European Systematic Risk Factor:

- **Market Integration**: 97.3% shared variance indicates extreme integration
  - European markets behave as single economic unit
  - Global economic conditions affect all markets simultaneously
  - ECB monetary policy, EU regulations, major political events

- **Portfolio Implications**:
  - Diversification within Europe provides minimal risk reduction
  - Need global (non-European) assets for meaningful diversification
  - European «diversified» portfolio = 97% systematic risk exposure
- **Risk Management**:
  - PC1 represents non-diversifiable risk within European context
  - PC2-PC4 (2.7% total) = market-specific idiosyncratic opportunities
  - Brexit effect visible in PC2 (FTSE vs continental separation)

Example 3A: Kuiper Belt Objects PCA

Kuiper Belt Objects: PCA Analysis

This section demonstrates PCA applied to astronomical data from the outer solar system.

- **Dataset**: Orbital parameters of 98 trans-Neptunian objects (TNOs) and Kuiper Belt objects
- **Research Question**: What are the main modes of orbital variation? Can we identify distinct dynamical populations?
- **Method**: Standardized PCA on 5 orbital elements with different physical units
- **Scripts**: `kuiper_pca.py`

Dataset: Orbital Parameters

Five key orbital elements describe each object’s motion:

- **a** (AU): Semi-major axis — average distance from Sun (30-150 AU)
- **e**: Eccentricity — orbital shape (0=circle, 1=parabola)
- **i** (degrees): Inclination — tilt relative to solar system plane
- **H** (magnitude): Absolute magnitude — brightness/size indicator

Known Dynamical Populations

Three main populations with distinct orbital signatures:

- **Classical Kuiper Belt** (60%): Low eccentricity, low inclination
  - Nearly circular orbits around 39-48 AU
  - «Cold» population — likely formed in place
- **Scattered Disk Objects** (30%): High eccentricity, distant
  - $e > 0.3$, semi-major axis > 50 AU
  - Scattered outward by gravitational encounters with Neptune
- **Resonant Objects** (10%): Locked in orbital resonances
  - 3:2 resonance at 39.4 AU (like Pluto)

PCA Results: Multi-Component Structure

Running `kuiper_pca.py` reveals distributed variance across components:

Component	Eigenvalue	% Variance	Cumulative %
PC1	2.009	39.8%	39.8%
PC2	1.079	21.4%	61.1%
PC3	1.036	20.5%	81.6%
PC4	0.628	12.4%	94.1%
PC5	0.299	5.9%	100.0%

- **Kaiser Criterion**: Retain PC1-PC3 (eigenvalues > 1.0)
- **Variance Distribution**: More balanced than previous examples
- **Interpretation**: Complex astronomical system requires multiple dimensions

Component Loadings: Astronomical Interpretation

Each component captures distinct aspects of orbital architecture:

Variable	PC1	PC2	PC3
a (distance)	0.571	−0.172	−0.578
e (eccentricity)	0.642	0.087	−0.117
i (inclination)	0.487	0.378	0.705

H (magnitude)	−0.157	0.905	−0.393

- **PC1**: «Orbital Excitation» - distance (a), eccentricity (e), inclination (i) correlate
- **PC2**: «Observational Bias» - brightness (H) dominates, reflecting size-distance effects
- **PC3**: «Resonant Structure» - inclination vs distance separation, identifies resonant families

PCA Interpretation: Dynamical Evolution

PC1 as Dynamical Excitation:

- High loadings on distance (a), eccentricity (e), and inclination (i)
- Represents gravitational «heating» of orbits over solar system history
- Separates pristine objects from those scattered by planetary migration
- **Implication**: Multiple gravitational processes create complex structure

Example 4A: Hospital Health Outcomes PCA

Hospital Health Outcomes: PCA Analysis

This section demonstrates PCA applied to healthcare quality data from US hospitals.

- **Dataset**: Health outcome metrics for 50 US hospitals across 8 performance indicators
- **Research Question**: What are the main dimensions of hospital quality? Can we rank hospital performance?
- **Method**: Standardized PCA on healthcare metrics with different units and scales
- **Scripts**: `hospitals_example.py`

Dataset: Hospital Performance Metrics

Eight key hospital quality indicators (with desired direction):

- **MortalityRate** (%): Hospital mortality rate (lower → better)
- **ReadmissionRate** (%): 30-day readmission rate (lower → better)
- **PatientSatisfaction** (0-100): Patient satisfaction score (higher → better)
- **AvgLengthStay** (days): Average length of stay (shorter → better)
- **InfectionRate** (%): Hospital-acquired infections (lower → better)
- **NurseRatio**: Nurse-to-patient ratio (higher → better)

- **SurgicalComplications** (%): Surgical complication rate (lower → better)
- **EDWaitTime** (minutes): Emergency dept. wait time (lower → better)

PCA Results: Strong Quality Factor

Running `hospitals_pca.py` reveals dominant quality dimension:

Component	Eigenvalue	% Variance	Cumulative %
PC1	5.752	70.5%	70.5%
PC2	0.695	8.5%	79.0%
PC3	0.480	5.9%	84.9%
PC4-PC8	all below 0.50	15.1%	100.0%

- **Dominant PC1**: Single quality factor explains 70.5% variance

- **Kaiser Criterion**: Only PC1 has eigenvalue > 1.0
- **Interpretation**: Hospital quality is largely unidimensional
- **Healthcare Insight**: Organizational excellence affects all metrics simultaneously

Component Loadings: Quality Halo Effect

PC1 shows consistent quality pattern across all metrics:

Health Outcome Metric	**PC1**	**PC2**	**PC3**
SurgicalComplications (↓)	−0.378	−0.002	−0.156
MortalityRate (↓)	−0.353	−0.368	−0.337
ReadmissionRate (↓)	−0.352	0.150	0.515
PatientSatisfaction (↑)	0.402	0.059	0.188
NurseRatio (↑)	0.381	0.041	0.160
InfectionRate (↓)	−0.337	0.187	−0.358

EDWaitTime (↓)	−0.311	−0.528	0.597
AvgLengthStay (↓)	−0.302	0.723	0.226

- **Consistent Pattern**: All «bad» outcomes load negatively, «good» outcomes positively
- **Quality Halo**: Excellent hospitals excel across all dimensions
- **PC2**: Efficiency dimension (length of stay vs wait time trade-off)

Healthcare Interpretation: Organizational Excellence

PC1 as Systematic Quality Factor:

- **Organizational Culture**: Leadership, processes, and culture affect all outcomes
  - Quality improvement programs create hospital-wide excellence
  - Poor management leads to problems across multiple domains
  - Safety culture and continuous improvement mindset crucial

- **Policy Implications**:
  - Hospital rankings can use single composite score (PC1)
  - Quality interventions should be comprehensive, not focused on single metrics
  - Resource allocation should target hospitals with low PC1 scores
- **Healthcare Economics**:
  - High-quality hospitals are more efficient (better outcomes at lower cost)
  - Bundled payments incentivize comprehensive quality improvement
  - Value-based care models align with PC1 structure