Appropriate Autoregressive Model Order

Introduction

The goal of this example is to find out the appropriate autoregressive (AR) model order using an algorithm based on the partial autocorrelation function (PAF). One acceleration time history from the baseline condition is used to carry out the analysis.

Data sets from **Channel 5** of the 3-story structure are used in this example. More details about the data sets can be found in the 3-Story Data Sets documentation.

The PAF-based algorithm suggests an AR model order as a reference starting point. Other algorithms should be tried in order to find out a possible range of AR model orders. (Note that the arModelOrder_shm function contains other techniques, namely, the SVD, AIC, BIC, and RMS.)

References:

Figueiredo, E., Park, G., Figueiras, J., Farrar, C., & Worden, K. (2009). Structural Health Monitoring Algorithm Comparisons using Standard Data Sets. Los Alamos National Laboratory Report: LA-14393.

SHMTools functions used:

- `ar_model_order_shm`

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
from pathlib import Path
import sys
import os

# Add shmtools to path - handle different execution contexts (lesson from pr
current_dir = Path.cwd()
notebook_dir = Path(__file__).parent if '__file__' in globals() else current

# Try different relative paths to find shmtools
possible_paths = [
    notebook_dir.parent.parent.parent,  # From examples/notebooks/basic/
    current_dir.parent.parent,          # From examples/notebooks/
    current_dir,                        # From project root
    Path('/Users/eric/repo/shm/shmtools-python')  # Absolute fallback
]

shmtools_found = False
for path in possible_paths:
    if (path / 'shmtools').exists():
        if str(path) not in sys.path:
            sys.path.insert(0, str(path))
        shmtools_found = True
        print(f"Found shmtools at: {path}")
        break

if not shmtools_found:
    print("Warning: Could not find shmtools module")

from shmtools.utils.data_loading import load_3story_data
from shmtools.features.time_series import ar_model_order_shm, ar_model_shm

# Set up plotting
plt.style.use('default')
plt.rcParams['figure.figsize'] = (12, 8)
plt.rcParams['font.size'] = 10
```

```
Found shmtools at: /Users/eric/repo/shm/shmtools-python
```

```
/Users/eric/repo/shm/shmtools-python/venv/lib/python3.9/site-packages/urllib
3/__init__.py:35: NotOpenSSLWarning: urllib3 v2 only supports OpenSSL 1.1.1
+, currently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: http
s://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
```

Load Raw Data

Load the 3-story structure dataset and extract Channel 5 data from a baseline condition for AR model order analysis.

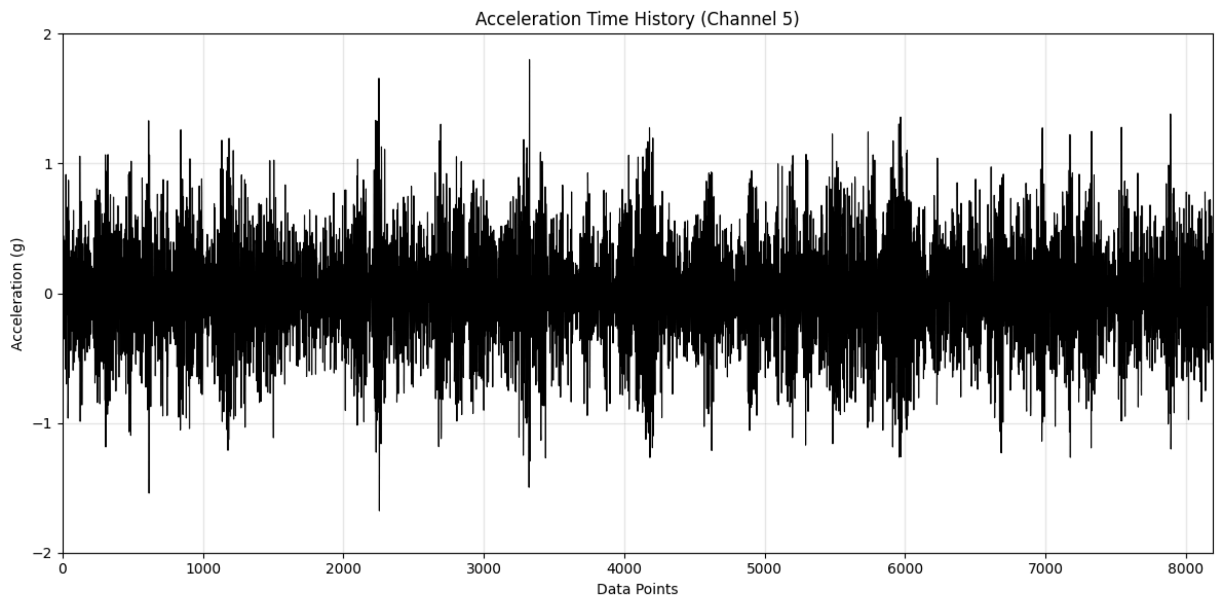
In [2]:
```
# Load data set
data_dict = load_3story_data()
dataset = data_dict['dataset']

print(f"Dataset shape: {dataset.shape}")

# Acceleration time history from the baseline condition (Channel 5)
data = dataset[:, 4, 0]  # Channel 5 (index 4), first condition (index 0)

print(f"Channel 5 baseline data shape: {data.shape}")
print(f"Mean: {np.mean(data):.6f}")
print(f"Std: {np.std(data):.6f}")
```

```
Dataset shape: (8192, 5, 170)
Channel 5 baseline data shape: (8192,)
Mean: -0.003130
Std: 0.409120
```

Plot Time History

Visualize the baseline acceleration time history from Channel 5.

In [3]:
```
# Plot time series
plt.figure(figsize=(12, 6))
plt.plot(data, 'k-', linewidth=0.8)
plt.title('Acceleration Time History (Channel 5)')
plt.xlabel('Data Points')
plt.ylabel('Acceleration (g)')
plt.xlim([0, len(data)])
plt.ylim([-2, 2])
plt.yticks([-2, -1, 0, 1, 2])
plt.grid(True, alpha=0.3)

plt.tight_layout()
plt.show()
```



Run Algorithm to find out the Appropriate AR Model Order

Using the PAF-based algorithm.

In [4]:
```
# Set parameters
method = 'PAF'
ar_order_max = 30
tolerance = 0.078

print(f"Running AR model order selection...")
print(f"Method: {method}")
print(f"Maximum order: {ar_order_max}")
print(f"Tolerance: {tolerance}")

# Run algorithm (following MATLAB exactly)
mean_ar_order, ar_orders, model = ar_model_order_shm(data, method, ar_order_

# Extract results from model structure
out_data = model['outData']
ar_order_list = model['arOrderList']
control_limit = model['controlLimit']
```

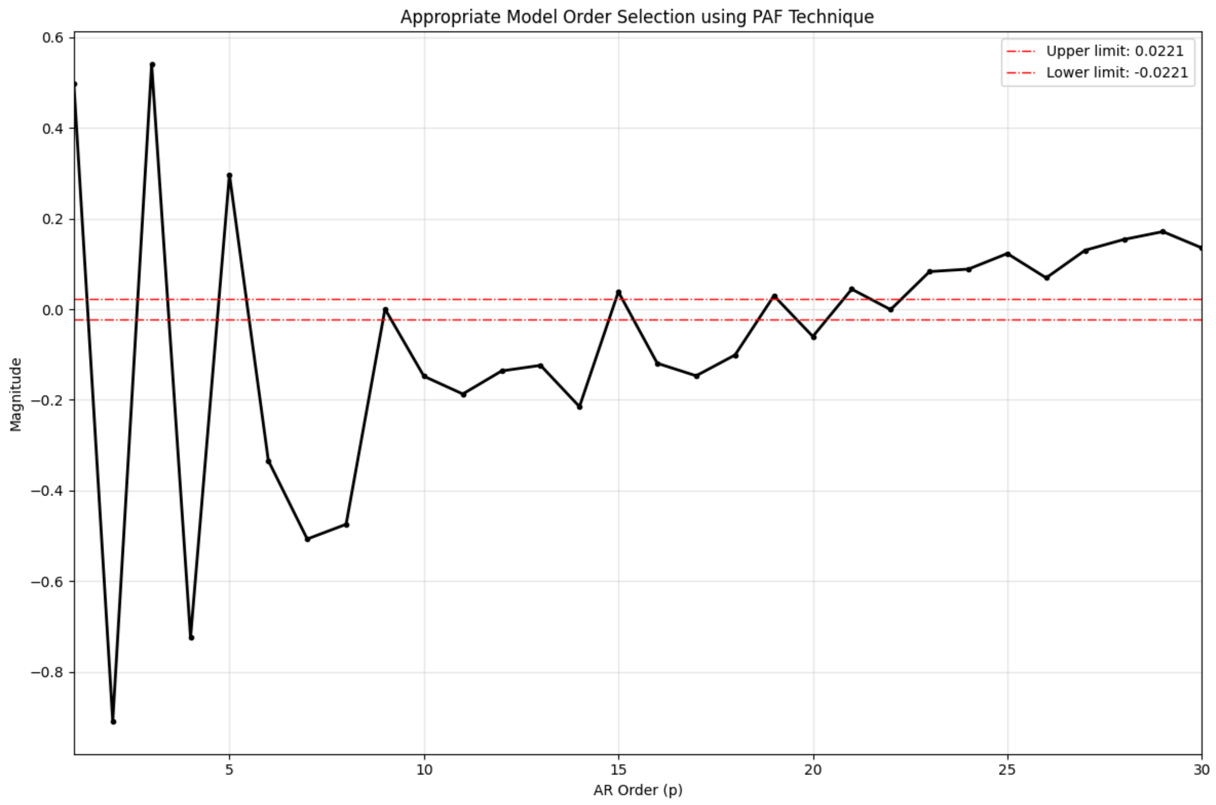
```
print(f"\nResults:")
print(f"Mean AR order: {mean_ar_order[0]:.0f}")
print(f"AR order for this instance: {ar_orders[0, 0]:.0f}")
print(f"Control limits: {control_limit}")
print(f"Method used: {model['method']}")
print(f"Maximum order computed: {model['arOrderMax']}")
print(f"Tolerance: {model['tolerance']}")
```

```
Running AR model order selection...
Method: PAF
Maximum order: 30
Tolerance: 0.078

Results:
Mean AR order: 9
AR order for this instance: 9
Control limits: [np.float64(0.022097086912079608), np.float64(-0.02209708691
2079608)]
Method used: PAF
Maximum order computed: 30
Tolerance: 0.078
```

Plot Results

Display the PAF values along with the confidence interval thresholds.

In [5]:
```
# Plot results with threshold (following MATLAB exactly)
plt.figure(figsize=(12, 8))

plt.plot(ar_order_list, out_data, '.-k', linewidth=2, markersize=6)
plt.title(f'Appropriate Model Order Selection using {method} Technique')
plt.xlabel('AR Order (p)')
plt.ylabel('Magnitude')
plt.xlim([1, max(ar_order_list)])
plt.grid(True, alpha=0.3)

# Add legend with AR order
plt.legend([f'AR Order: {int(mean_ar_order[0])}'])

# Add control limit lines
plt.axhline(y=control_limit[0], color='r', linestyle='-.', linewidth=1,
           label=f'Upper limit: {control_limit[0]:.4f}')

if method == 'PAF':
    plt.axhline(y=control_limit[1], color='r', linestyle='-.', linewidth=1,
               label=f'Lower limit: {control_limit[1]:.4f}')

plt.legend()
plt.tight_layout()
plt.show()

# Summary message (following MATLAB)
print(f"\nThe {method}-based algorithm suggests an AR model of {int(mean_ar_
print("This indication should be taken as a reference for a starting point."
print("Other algorithms should be tried in order to find out a possible rang
print("AR model orders. (Note that the arModelOrder_shm function contains ot
print("techniques, namely, the SVD, AIC, BIC, and RMS.)")
```



```
The PAF-based algorithm suggests an AR model of 9th order.
This indication should be taken as a reference for a starting point.
Other algorithms should be tried in order to find out a possible range of
AR model orders. (Note that the arModelOrder_shm function contains other
techniques, namely, the SVD, AIC, BIC, and RMS.)
```

Summary

This example demonstrated AR model order selection using the Partial Autocorrelation Function (PAF) method. The algorithm suggests an AR model order by finding the first order where the PAF value falls within the confidence bounds for white noise.

Key Results:

- The PAF method suggested an AR order based on 95% confidence intervals ($\pm 2/\sqrt{N}$)
- This provides a starting point for AR model selection in SHM applications
- Other methods (AIC, BIC, SVD, RMS) are available in the `ar_model_order_shm` function for comparison

For Structural Health Monitoring:

The choice of AR order affects the quality of damage-sensitive features extracted from time series data. This systematic approach to order selection helps ensure that AR models capture the essential dynamics while avoiding overfitting.

See also:

- Outlier Detection based on Principal Component Analysis
- Outlier Detection based on Mahalanobis Distance
- Outlier Detection based on Singular Value Decomposition
- Outlier Detection based on Factor Analysis

Visualize Model Predictions and Residuals

Compare the prediction accuracy and residual patterns for different AR model orders.