01-metro-housing-wage-divergence

November 28, 2025

1 Metro Housing-Wage Divergence Analysis

Analyzing the Growing Gap Between Housing Costs and Wages Using Real Economic Data

---

1.1 Executive Summary

This notebook demonstrates how to analyze the divergence between housing costs and wages using the **KRL Suite** - specifically the `krl-data-connectors` package to fetch real economic data from FRED (Federal Reserve Economic Data) and BLS (Bureau of Labor Statistics).

1.1.1 KRL Suite Components Used

- **krl_data_connectors.community**: `FREDBasicConnector`, `BLSBasicConnector` for real-time economic data
- **krl_core**: `get_logger` for structured logging

1.1.2 What You'll Learn

1. Fetching housing and wage data from FRED and BLS using KRL connectors
2. Computing divergence metrics between housing costs and wages
3. Visualizing temporal patterns in housing affordability
4. Understanding the gap between housing and wage growth

Estimated Time: 15-20 minutes
Difficulty: Beginner to Intermediate

> **Note:** This notebook uses the Community tier connectors which provide free access to national-level economic data without API keys.

1.2 Table of Contents

1. Setup and Imports
2. Data Loading
3. Exploratory Analysis
4. Divergence Calculation
5. Visualization
6. Key Insights
7. Next Steps

8. Data Provenance

1. Setup and Imports

First, we'll import the required libraries and configure the demo environment.

[20]:
```
# Standard library imports
import os
import sys
import warnings
from datetime import datetime
import importlib

# Add KRL package paths (handles spaces in path correctly)
_krl_base = os.path.expanduser("~/Documents/GitHub/KRL/Private IP")
for _pkg in ["krl-open-core/src", "krl-data-connectors/src"]:
    _path = os.path.join(_krl_base, _pkg)
    if _path not in sys.path:
        sys.path.insert(0, _path)

# Load environment variables from .env file
from dotenv import load_dotenv
_env_path = os.path.expanduser("~/Documents/GitHub/KRL/krl-tutorials/.env")
load_dotenv(_env_path)

# Force complete reload of KRL modules to pick up any changes
_modules_to_reload = [k for k in sys.modules.keys() if k.
 ↪startswith(('krl_core', 'krl_data_connectors'))]
for _mod in _modules_to_reload:
    del sys.modules[_mod]

# Data manipulation
import pandas as pd
import numpy as np

# Visualization
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots

# ============================================================================
# KRL Suite Imports - These are the REAL package imports
# ============================================================================

# KRL Data Connectors - Community Tier (Free, no API key required)
from krl_data_connectors.community import (
    FREDBasicConnector,     # Federal Reserve Economic Data
    BLSBasicConnector,      # Bureau of Labor Statistics
```

```
)

# KRL Core - Logging and utilities
from krl_core import get_logger

# Configure display
pd.set_option('display.max_columns', 20)
pd.set_option('display.float_format', '{:,.2f}'.format)
warnings.filterwarnings('ignore', category=FutureWarning)

# Initialize logger
logger = get_logger("HousingWageDivergence")

# Session info
print("=" * 60)
print("  Metro Housing-Wage Divergence Analysis")
print("=" * 60)
print(f"  Execution Time: {datetime.now().strftime('%Y-%m-%d %H:%M:%S')}")
print(f"  Using KRL Data Connectors (Community Tier)")
print(f"  FRED API Key: {'  Loaded' if os.getenv('FRED_API_KEY') else '  Not␣
  ↪found'}")
print("=" * 60)
```

```
============================================================
  Metro Housing-Wage Divergence Analysis
============================================================
  Execution Time: 2025-11-27 11:44:06
  Using KRL Data Connectors (Community Tier)
  FRED API Key:   Loaded
============================================================
```

2. Data Loading

We'll use the **KRL Data Connectors** to fetch real economic data:

1. **FREDBasicConnector**: Housing starts (HOUST) as a proxy for housing market activity
2. **BLSBasicConnector**: Average hourly earnings and employment data

   Community Tier Benefits: - No API key required - Access to national-level economic indicators - Up to 10 years of historical data

[21]:
```
# ================================================================================
# Initialize KRL Data Connectors
# ================================================================================

# Initialize FRED connector (Federal Reserve Economic Data)
fred = FREDBasicConnector()

# Initialize BLS connector (Bureau of Labor Statistics)
```

```python
bls = BLSBasicConnector()

# Test connections
print("  Testing API Connections...")
print(f"   FRED Connected: {fred.connect()}")
print(f"   BLS Connected: {bls.connect()}")

# List available series
print("\n  Available FRED Series (Community Tier):")
for series_id, description in list(fred.AVAILABLE_SERIES.items())[:5]:
    print(f"   • {series_id}: {description}")
print("   ...")

print("\n  Available BLS Series (Community Tier):")
for series_id, description in list(bls.AVAILABLE_SERIES.items())[:5]:
    print(f"   • {series_id}: {description}")
print("   ...")
```


```
{"timestamp": "2025-11-27T16:44:14.418098Z", "level": "INFO", "name":
"FREDBasicConnector", "message": "Connector initialized", "source": {"file":
"base_connector.py", "line": 81, "function": "__init__"}, "levelname": "INFO",
"taskName": "Task-93", "connector": "FREDBasicConnector", "cache_dir":
"/Users/bcdelo/.krl_cache/fredbasicconnector", "cache_ttl": 3600, "has_api_key":
true}
{"timestamp": "2025-11-27T16:44:14.418575Z", "level": "INFO", "name":
"FREDBasicConnector", "message": "Initialized FRED Basic connector (Community
tier)", "source": {"file": "fred_basic.py", "line": 96, "function": "__init__"},
"levelname": "INFO", "taskName": "Task-93", "available_series": 15}
{"timestamp": "2025-11-27T16:44:14.419310Z", "level": "WARNING", "name":
"BLSBasicConnector", "message": "No API key provided", "source": {"file":
"base_connector.py", "line": 74, "function": "__init__"}, "levelname":
"WARNING", "taskName": "Task-93", "connector": "BLSBasicConnector"}
{"timestamp": "2025-11-27T16:44:14.420032Z", "level": "INFO", "name":
"BLSBasicConnector", "message": "Connector initialized", "source": {"file":
"base_connector.py", "line": 81, "function": "__init__"}, "levelname": "INFO",
"taskName": "Task-93", "connector": "BLSBasicConnector", "cache_dir":
"/Users/bcdelo/.krl_cache/blsbasicconnector", "cache_ttl": 3600, "has_api_key":
false}
{"timestamp": "2025-11-27T16:44:14.420334Z", "level": "INFO", "name":
"BLSBasicConnector", "message": "Initialized BLS Basic connector (Community
tier)", "source": {"file": "bls_basic.py", "line": 89, "function": "__init__"},
"levelname": "INFO", "taskName": "Task-93", "available_series": 8}
{"timestamp": "2025-11-27T16:44:14.418575Z", "level": "INFO", "name":
"FREDBasicConnector", "message": "Initialized FRED Basic connector (Community
tier)", "source": {"file": "fred_basic.py", "line": 96, "function": "__init__"},
"levelname": "INFO", "taskName": "Task-93", "available_series": 15}
{"timestamp": "2025-11-27T16:44:14.419310Z", "level": "WARNING", "name":
"BLSBasicConnector", "message": "No API key provided", "source": {"file":
```

"base_connector.py", "line": 74, "function": "__init__"}, "levelname":
"WARNING", "taskName": "Task-93", "connector": "BLSBasicConnector"}
{"timestamp": "2025-11-27T16:44:14.420032Z", "level": "INFO", "name":
"BLSBasicConnector", "message": "Connector initialized", "source": {"file":
"base_connector.py", "line": 81, "function": "__init__"}, "levelname": "INFO",
"taskName": "Task-93", "connector": "BLSBasicConnector", "cache_dir":
"/Users/bcdelo/.krl_cache/blsbasicconnector", "cache_ttl": 3600, "has_api_key":
false}
{"timestamp": "2025-11-27T16:44:14.420334Z", "level": "INFO", "name":
"BLSBasicConnector", "message": "Initialized BLS Basic connector (Community
tier)", "source": {"file": "bls_basic.py", "line": 89, "function": "__init__"},
"levelname": "INFO", "taskName": "Task-93", "available_series": 8}
 Testing API Connections…
{"timestamp": "2025-11-27T16:44:14.627609Z", "level": "INFO", "name":
"FREDBasicConnector", "message": "Successfully connected to FRED API", "source":
{"file": "fred_basic.py", "line": 131, "function": "connect"}, "levelname":
"INFO", "taskName": "Task-93"}
 Testing API Connections…
{"timestamp": "2025-11-27T16:44:14.627609Z", "level": "INFO", "name":
"FREDBasicConnector", "message": "Successfully connected to FRED API", "source":
{"file": "fred_basic.py", "line": 131, "function": "connect"}, "levelname":
"INFO", "taskName": "Task-93"}
   FRED Connected: True
   FRED Connected: True
{"timestamp": "2025-11-27T16:44:14.996239Z", "level": "INFO", "name":
"BLSBasicConnector", "message": "Successfully connected to BLS API", "source":
{"file": "bls_basic.py", "line": 128, "function": "connect"}, "levelname":
"INFO", "taskName": "Task-93"}
   BLS Connected: True

 Available FRED Series (Community Tier):
   • UNRATE: Unemployment Rate
   • GDP: Gross Domestic Product
   • CPIAUCSL: Consumer Price Index for All Urban Consumers
   • FEDFUNDS: Federal Funds Effective Rate
   • DGS10: 10-Year Treasury Constant Maturity Rate
   …

 Available BLS Series (Community Tier):
   • LNS14000000: Unemployment Rate (National)
   • LNS12000000: Employment Level (National)
   • LNS11000000: Civilian Labor Force (National)
   • CUUR0000SA0: CPI-U All Items (National)
   • CUUR0000SAF: CPI-U Food (National)
   …
{"timestamp": "2025-11-27T16:44:14.996239Z", "level": "INFO", "name":
"BLSBasicConnector", "message": "Successfully connected to BLS API", "source":
{"file": "bls_basic.py", "line": 128, "function": "connect"}, "levelname":

```
"INFO", "taskName": "Task-93"}
   BLS Connected: True

  Available FRED Series (Community Tier):
    • UNRATE: Unemployment Rate
    • GDP: Gross Domestic Product
    • CPIAUCSL: Consumer Price Index for All Urban Consumers
    • FEDFUNDS: Federal Funds Effective Rate
    • DGS10: 10-Year Treasury Constant Maturity Rate
    …

  Available BLS Series (Community Tier):
    • LNS14000000: Unemployment Rate (National)
    • LNS12000000: Employment Level (National)
    • LNS11000000: Civilian Labor Force (National)
    • CUUR0000SA0: CPI-U All Items (National)
    • CUUR0000SAF: CPI-U Food (National)
    …
```

[22]:
```python
# ============================================================================
# Fetch Housing Market Data from FRED
# ============================================================================

# Housing Starts (HOUST) - New residential construction
try:
    housing_df = fred.get_series("HOUST", start_date="2015-01-01",␣
 ↪end_date="2024-12-31")
    print(" Loaded live data from FRED API")
except Exception as e:
    # Fallback to synthetic demo data for showcase
    print(f" FRED API unavailable ({type(e).__name__}), using demo data...")
    dates = pd.date_range("2015-01-01", "2024-12-01", freq="MS")
    # Realistic housing starts pattern (thousands of units)
    np.random.seed(42)
    base = 1100 + np.linspace(0, 300, len(dates))
    seasonal = 100 * np.sin(np.linspace(0, 20*np.pi, len(dates)))
    covid_dip = np.where((dates >= "2020-03-01") & (dates <= "2020-06-01"),␣
 ↪-300, 0)
    values = base + seasonal + covid_dip + np.random.normal(0, 30, len(dates))
    housing_df = pd.DataFrame({"value": values}, index=dates)

print(" Housing Starts Data (HOUST):")
print(f"   Shape: {housing_df.shape}")
print(f"   Date Range: {housing_df.index.min()} to {housing_df.index.max()}")
print(f"\n   Preview:")
housing_df.head()
```

```
{"timestamp": "2025-11-27T16:44:21.820457Z", "level": "INFO", "name":
```

"FREDBasicConnector", "message": "Fetching FRED series: HOUST", "source":
{"file": "fred_basic.py", "line": 167, "function": "get_series"}, "levelname":
"INFO", "taskName": "Task-96", "series_id": "HOUST", "start_date": "2015-01-01",
"end_date": "2024-12-31"}
{"timestamp": "2025-11-27T16:44:21.959778Z", "level": "INFO", "name":
"FREDBasicConnector", "message": "Retrieved 120 observations for HOUST",
"source": {"file": "fred_basic.py", "line": 197, "function": "get_series"},
"levelname": "INFO", "taskName": "Task-96", "series_id": "HOUST", "rows": 120}
 Loaded live data from FRED API
 Housing Starts Data (HOUST):
   Shape: (120, 1)
   Date Range: 2015-01-01 00:00:00 to 2024-12-01 00:00:00

   Preview:
{"timestamp": "2025-11-27T16:44:21.959778Z", "level": "INFO", "name":
"FREDBasicConnector", "message": "Retrieved 120 observations for HOUST",
"source": {"file": "fred_basic.py", "line": 197, "function": "get_series"},
"levelname": "INFO", "taskName": "Task-96", "series_id": "HOUST", "rows": 120}
 Loaded live data from FRED API
 Housing Starts Data (HOUST):
   Shape: (120, 1)
   Date Range: 2015-01-01 00:00:00 to 2024-12-01 00:00:00

   Preview:

[22]:
date	value
2015-01-01	1,085.00
2015-02-01	886.00
2015-03-01	960.00
2015-04-01	1,190.00
2015-05-01	1,079.00

[23]:
```
# ============================================================================
# Fetch Wage Data from BLS
# ============================================================================

# Average Hourly Earnings (National)
earnings_df = bls.get_series("CES0500000003")  # Average Hourly Earnings␣
 ↪(National)
print("  Average Hourly Earnings Data (BLS):")
print(f"   Shape: {earnings_df.shape}")
print(f"   Date Range: {earnings_df.index.min()} to {earnings_df.index.max()}")
print(f"\n   Preview:")
earnings_df.head()
```

{"timestamp": "2025-11-27T16:44:52.313165Z", "level": "INFO", "name":
"BLSBasicConnector", "message": "Fetching BLS series: CES0500000003", "source":

```
{"file": "bls_basic.py", "line": 196, "function": "get_series"}, "levelname":
"INFO", "taskName": "Task-99", "series_id": "CES0500000003", "start_year": 2016,
"end_year": 2025}
{"timestamp": "2025-11-27T16:44:52.449010Z", "level": "INFO", "name":
"BLSBasicConnector", "message": "Retrieved 117 observations for CES0500000003",
"source": {"file": "bls_basic.py", "line": 242, "function": "get_series"},
"levelname": "INFO", "taskName": "Task-99", "series_id": "CES0500000003",
"rows": 117}
 Average Hourly Earnings Data (BLS):
   Shape: (117, 6)
   Date Range: 2016-01-01 00:00:00 to 2025-09-01 00:00:00

   Preview:
{"timestamp": "2025-11-27T16:44:52.449010Z", "level": "INFO", "name":
"BLSBasicConnector", "message": "Retrieved 117 observations for CES0500000003",
"source": {"file": "bls_basic.py", "line": 242, "function": "get_series"},
"levelname": "INFO", "taskName": "Task-99", "series_id": "CES0500000003",
"rows": 117}
 Average Hourly Earnings Data (BLS):
   Shape: (117, 6)
   Date Range: 2016-01-01 00:00:00 to 2025-09-01 00:00:00

   Preview:
```

[23]:

date	year	period	periodName	latest	value	footnotes
2016-01-01	2016	M01	January	NaN	25.37	[{}]
2016-02-01	2016	M02	February	NaN	25.38	[{}]
2016-03-01	2016	M03	March	NaN	25.45	[{}]
2016-04-01	2016	M04	April	NaN	25.53	[{}]
2016-05-01	2016	M05	May	NaN	25.58	[{}]

3. Exploratory Analysis

Let's also fetch additional economic indicators to understand the broader economic context: - **CPI (CPIAUCSL)**: Consumer Price Index to measure inflation - **Unemployment Rate (LNS14000000)**: Labor market health

[24]:
```
# ============================================================================
# Fetch Additional Economic Indicators
# ============================================================================

# Consumer Price Index (inflation measure)
cpi_df = fred.get_series("CPIAUCSL", start_date="2015-01-01",␣
 ↪end_date="2024-12-31")
print("  Consumer Price Index (CPI):")
print(f"   Shape: {cpi_df.shape}")
```

```python
# Unemployment Rate from BLS
unemployment_df = bls.get_unemployment_rate()
print(f"\n Unemployment Rate:")
print(f"   Shape: {unemployment_df.shape}")

# Mortgage Rates from FRED
mortgage_df = fred.get_series("MORTGAGE30US", start_date="2015-01-01",␣
 ↪end_date="2024-12-31")
print(f"\n 30-Year Mortgage Rate:")
print(f"   Shape: {mortgage_df.shape}")
```

```
{"timestamp": "2025-11-27T16:44:52.457416Z", "level": "INFO", "name":
"FREDBasicConnector", "message": "Fetching FRED series: CPIAUCSL", "source":
{"file": "fred_basic.py", "line": 167, "function": "get_series"}, "levelname":
"INFO", "taskName": "Task-102", "series_id": "CPIAUCSL", "start_date":
"2015-01-01", "end_date": "2024-12-31"}
{"timestamp": "2025-11-27T16:44:52.673654Z", "level": "INFO", "name":
"FREDBasicConnector", "message": "Retrieved 120 observations for CPIAUCSL",
"source": {"file": "fred_basic.py", "line": 197, "function": "get_series"},
"levelname": "INFO", "taskName": "Task-102", "series_id": "CPIAUCSL", "rows":
120}
 Consumer Price Index (CPI):
   Shape: (120, 1)
{"timestamp": "2025-11-27T16:44:52.674142Z", "level": "INFO", "name":
"BLSBasicConnector", "message": "Fetching BLS series: LNS14000000", "source":
{"file": "bls_basic.py", "line": 196, "function": "get_series"}, "levelname":
"INFO", "taskName": "Task-102", "series_id": "LNS14000000", "start_year": 2016,
"end_year": 2025}
{"timestamp": "2025-11-27T16:44:52.673654Z", "level": "INFO", "name":
"FREDBasicConnector", "message": "Retrieved 120 observations for CPIAUCSL",
"source": {"file": "fred_basic.py", "line": 197, "function": "get_series"},
"levelname": "INFO", "taskName": "Task-102", "series_id": "CPIAUCSL", "rows":
120}
 Consumer Price Index (CPI):
   Shape: (120, 1)
{"timestamp": "2025-11-27T16:44:52.674142Z", "level": "INFO", "name":
"BLSBasicConnector", "message": "Fetching BLS series: LNS14000000", "source":
{"file": "bls_basic.py", "line": 196, "function": "get_series"}, "levelname":
"INFO", "taskName": "Task-102", "series_id": "LNS14000000", "start_year": 2016,
"end_year": 2025}
{"timestamp": "2025-11-27T16:44:52.807739Z", "level": "INFO", "name":
"BLSBasicConnector", "message": "Retrieved 117 observations for LNS14000000",
"source": {"file": "bls_basic.py", "line": 242, "function": "get_series"},
"levelname": "INFO", "taskName": "Task-102", "series_id": "LNS14000000", "rows":
117}

 Unemployment Rate:
   Shape: (117, 6)
```

```
{"timestamp": "2025-11-27T16:44:52.808312Z", "level": "INFO", "name":
"FREDBasicConnector", "message": "Fetching FRED series: MORTGAGE30US", "source":
{"file": "fred_basic.py", "line": 167, "function": "get_series"}, "levelname":
"INFO", "taskName": "Task-102", "series_id": "MORTGAGE30US", "start_date":
"2015-01-01", "end_date": "2024-12-31"}
{"timestamp": "2025-11-27T16:44:52.807739Z", "level": "INFO", "name":
"BLSBasicConnector", "message": "Retrieved 117 observations for LNS14000000",
"source": {"file": "bls_basic.py", "line": 242, "function": "get_series"},
"levelname": "INFO", "taskName": "Task-102", "series_id": "LNS14000000", "rows":
117}

 Unemployment Rate:
   Shape: (117, 6)
{"timestamp": "2025-11-27T16:44:52.808312Z", "level": "INFO", "name":
"FREDBasicConnector", "message": "Fetching FRED series: MORTGAGE30US", "source":
{"file": "fred_basic.py", "line": 167, "function": "get_series"}, "levelname":
"INFO", "taskName": "Task-102", "series_id": "MORTGAGE30US", "start_date":
"2015-01-01", "end_date": "2024-12-31"}
{"timestamp": "2025-11-27T16:44:53.039949Z", "level": "INFO", "name":
"FREDBasicConnector", "message": "Retrieved 521 observations for MORTGAGE30US",
"source": {"file": "fred_basic.py", "line": 197, "function": "get_series"},
"levelname": "INFO", "taskName": "Task-102", "series_id": "MORTGAGE30US",
"rows": 521}

 30-Year Mortgage Rate:
   Shape: (521, 1)
{"timestamp": "2025-11-27T16:44:53.039949Z", "level": "INFO", "name":
"FREDBasicConnector", "message": "Retrieved 521 observations for MORTGAGE30US",
"source": {"file": "fred_basic.py", "line": 197, "function": "get_series"},
"levelname": "INFO", "taskName": "Task-102", "series_id": "MORTGAGE30US",
"rows": 521}

 30-Year Mortgage Rate:
   Shape: (521, 1)
```

[25]:
```python
# ============================================================================
# Merge and Prepare Data
# ============================================================================

# Resample all series to monthly for consistency
def prepare_series(df, column_name):
    """Prepare a series with a named column."""
    result = df[['value']].copy()
    result.columns = [column_name]
    return result

# Prepare each dataset
```

```
housing = prepare_series(housing_df, 'housing_starts')
earnings = prepare_series(earnings_df, 'avg_hourly_earnings')
cpi = prepare_series(cpi_df, 'cpi')
mortgage = prepare_series(mortgage_df, 'mortgage_rate')

# Merge all series on date index
combined_df = housing.join([earnings, cpi, mortgage], how='outer')

# Forward-fill missing values (BLS is monthly, FRED is weekly for some series)
combined_df = combined_df.resample('MS').first().dropna()

print(" Combined Economic Dataset:")
print(f"   Shape: {combined_df.shape}")
print(f"   Date Range: {combined_df.index.min()} to {combined_df.index.max()}")
print(f"   Columns: {list(combined_df.columns)}")
combined_df.head(10)
```

```
 Combined Economic Dataset:
   Shape: (108, 4)
   Date Range: 2016-01-01 00:00:00 to 2024-12-01 00:00:00
   Columns: ['housing_starts', 'avg_hourly_earnings', 'cpi', 'mortgage_rate']
```

[25]:

date	housing_starts	avg_hourly_earnings	cpi	mortgage_rate
2016-01-01	1,092.00	25.37	237.65	3.97
2016-02-01	1,225.00	25.38	237.34	3.72
2016-03-01	1,111.00	25.45	238.08	3.64
2016-04-01	1,163.00	25.53	238.99	3.59
2016-05-01	1,148.00	25.58	239.56	3.61
2016-06-01	1,203.00	25.62	240.22	3.66
2016-07-01	1,239.00	25.69	240.10	3.41
2016-08-01	1,171.00	25.71	240.54	3.43
2016-09-01	1,068.00	25.77	241.18	3.46
2016-10-01	1,313.00	25.90	241.74	3.42

4. Divergence Calculation

The **divergence index** measures how much faster housing costs (proxied by housing market activity and mortgage rates) have grown compared to wages. We'll calculate:

1. **Cumulative Growth Rates**: How much each indicator has grown since baseline
2. **Real vs Nominal**: Adjust wages for inflation using CPI
3. **Affordability Index**: Housing cost burden relative to wages

[26]:
```
# ============================================================================
# Calculate Divergence Metrics
# ============================================================================

# Get baseline values (first observation)
```

```
baseline = combined_df.iloc[0]

# Calculate cumulative growth rates (%)
growth_df = combined_df.copy()
for col in combined_df.columns:
    growth_df[f'{col}_growth'] = ((combined_df[col] / baseline[col]) - 1) * 100

# Calculate real wage growth (adjusted for inflation)
cpi_baseline = baseline['cpi']
growth_df['cpi_multiplier'] = cpi_baseline / combined_df['cpi']
growth_df['real_earnings'] = combined_df['avg_hourly_earnings'] *␣
 ↪growth_df['cpi_multiplier']
growth_df['real_earnings_growth'] = ((growth_df['real_earnings'] /␣
 ↪baseline['avg_hourly_earnings']) - 1) * 100

# Calculate wage-housing divergence
growth_df['nominal_divergence'] = growth_df['cpi_growth'] -␣
 ↪growth_df['avg_hourly_earnings_growth']
growth_df['housing_wage_ratio'] = combined_df['mortgage_rate'] /␣
 ↪combined_df['avg_hourly_earnings']

print(" Growth Metrics Calculated:")
print(f"   Total observations: {len(growth_df)}")
print(f"\n   Latest Values (as of {growth_df.index[-1].strftime('%Y-%m')}):")
print(f"   • Wage Growth (Nominal): {growth_df['avg_hourly_earnings_growth'].
 ↪iloc[-1]:.1f}%")
print(f"   • Wage Growth (Real): {growth_df['real_earnings_growth'].iloc[-1]:.
 ↪1f}%")
print(f"   • Inflation (CPI): {growth_df['cpi_growth'].iloc[-1]:.1f}%")
print(f"   • Mortgage Rate: {combined_df['mortgage_rate'].iloc[-1]:.2f}%")
```

```
 Growth Metrics Calculated:
   Total observations: 108

   Latest Values (as of 2024-12):
   • Wage Growth (Nominal): 40.6%
   • Wage Growth (Real): 5.2%
   • Inflation (CPI): 33.6%
   • Mortgage Rate: 6.69%
```

[27]:
```
# ============================================================================
# Summary Statistics by Year
# ============================================================================

# Aggregate to annual for clearer trends
annual_df = growth_df.resample('YS').mean()
```

```
# Calculate year-over-year changes
annual_summary = pd.DataFrame({
    'Year': annual_df.index.year,
    'Avg Hourly Earnings ($)': annual_df['avg_hourly_earnings'].round(2),
    'Wage Growth (Nominal %)': annual_df['avg_hourly_earnings_growth'].round(1),
    'Wage Growth (Real %)': annual_df['real_earnings_growth'].round(1),
    'CPI Growth (%)': annual_df['cpi_growth'].round(1),
    'Mortgage Rate (%)': annual_df['mortgage_rate'].round(2),
    'Housing Starts (000s)': annual_df['housing_starts'].round(0),
})
annual_summary = annual_summary.set_index('Year')

print("  Annual Economic Summary:")
annual_summary
```

```
 Annual Economic Summary:
```

[27]:

Year	Avg Hourly Earnings ($)	Wage Growth (Nominal %)	Wage Growth (Real %)	CPI Growth (%)	Mortgage Rate (%)	Housing Starts (000s)
2016	25.65	1.10	0.10	1.00	3.63	1,177.00
2017	26.31	3.70	0.50	3.10	4.00	1,205.00
2018	27.10	6.80	1.10	5.70	4.50	1,247.00
2019	28.00	10.40	2.60	7.60	3.95	1,292.00
2020	29.36	15.70	6.30	8.90	3.12	1,394.00
2021	30.62	20.70	5.90	14.00	2.94	1,603.00
2022	32.27	27.20	3.30	23.10	5.14	1,552.00
2023	33.70	32.80	3.60	28.20	6.82	1,421.00
2024	35.06	38.20	4.70	32.00	6.73	1,371.00

5. Visualization

Let's create visualizations to understand the patterns in the data using the real economic indicators we fetched from FRED and BLS.

[28]:
```
# ============================================================================
# Visualization 1: Wage Growth vs Inflation
# ============================================================================
```

```
fig = go.Figure()

# Nominal wage growth
fig.add_trace(go.Scatter(
    x=growth_df.index,
    y=growth_df['avg_hourly_earnings_growth'],
    name='Nominal Wage Growth',
    line=dict(color='#0077BB', width=2),
))

# Real wage growth
fig.add_trace(go.Scatter(
    x=growth_df.index,
    y=growth_df['real_earnings_growth'],
    name='Real Wage Growth (Inflation-Adjusted)',
    line=dict(color='#009988', width=2),
))

# Inflation (CPI)
fig.add_trace(go.Scatter(
    x=growth_df.index,
    y=growth_df['cpi_growth'],
    name='Inflation (CPI)',
    line=dict(color='#EE7733', width=2, dash='dash'),
))

fig.update_layout(
    title='Wage Growth vs Inflation: Has Pay Kept Up with Prices?',
    xaxis_title='Date',
    yaxis_title='Cumulative Growth (%)',
    template='plotly_white',
    height=500,
    legend=dict(yanchor="top", y=0.99, xanchor="left", x=0.01),
    hovermode='x unified',
)

# Add zero line
fig.add_hline(y=0, line_dash="dash", line_color="gray", opacity=0.5)

fig.show()
```

[29]:
```
# ============================================================================
# Visualization 2: Housing Market Indicators
# ============================================================================

fig = make_subplots(
```

```
    rows=2, cols=1,
    subplot_titles=('Housing Starts (New Residential Construction)', '30-Year␣
 ↪Mortgage Rate'),
    vertical_spacing=0.12,
)

# Housing starts
fig.add_trace(
    go.Scatter(
        x=combined_df.index,
        y=combined_df['housing_starts'],
        name='Housing Starts',
        fill='tozeroy',
        fillcolor='rgba(0, 119, 187, 0.2)',
        line=dict(color='#0077BB', width=2),
    ),
    row=1, col=1
)

# Mortgage rate
fig.add_trace(
    go.Scatter(
        x=combined_df.index,
        y=combined_df['mortgage_rate'],
        name='30-Year Mortgage Rate',
        line=dict(color='#CC3311', width=2),
    ),
    row=2, col=1
)

fig.update_layout(
    title='Housing Market Conditions: Supply and Financing Costs',
    template='plotly_white',
    height=600,
    showlegend=False,
)

fig.update_yaxes(title_text="Units (Thousands)", row=1, col=1)
fig.update_yaxes(title_text="Rate (%)", row=2, col=1)

fig.show()
```

[30]:
```
# ============================================================================
# Visualization 3: Economic Dashboard
# ============================================================================

fig = make_subplots(
```

```
    rows=2, cols=2,
    subplot_titles=(
        'Average Hourly Earnings ($)',
        'Consumer Price Index',
        'Housing Starts (000s)',
        'Mortgage Rate (%)'
    ),
    vertical_spacing=0.12,
    horizontal_spacing=0.08,
)

# Wages
fig.add_trace(
    go.Scatter(x=combined_df.index, y=combined_df['avg_hourly_earnings'],
               line=dict(color='#0077BB'), name='Earnings'),
    row=1, col=1
)

# CPI
fig.add_trace(
    go.Scatter(x=combined_df.index, y=combined_df['cpi'],
               line=dict(color='#009988'), name='CPI'),
    row=1, col=2
)

# Housing Starts
fig.add_trace(
    go.Scatter(x=combined_df.index, y=combined_df['housing_starts'],
               line=dict(color='#EE7733'), name='Housing'),
    row=2, col=1
)

# Mortgage Rate
fig.add_trace(
    go.Scatter(x=combined_df.index, y=combined_df['mortgage_rate'],
               line=dict(color='#CC3311'), name='Mortgage'),
    row=2, col=2
)

fig.update_layout(
    title='Economic Indicators Dashboard (FRED + BLS Data)',
    template='plotly_white',
    height=600,
    showlegend=False,
)

fig.show()
```

[32]:
```
# ============================================================================
# Visualization 4: Affordability Stress Index
# ============================================================================

# Calculate an affordability stress index
# Higher mortgage rates + lower wages = more stress
# Normalize both to baseline and combine

mortgage_stress = combined_df['mortgage_rate'] / baseline['mortgage_rate']
wage_relief = combined_df['avg_hourly_earnings'] /␣
 ↪baseline['avg_hourly_earnings']

# Stress index: mortgage stress / wage relief
# Values > 1 mean affordability is worse than baseline
growth_df['affordability_stress'] = (mortgage_stress / wage_relief) * 100

fig = go.Figure()

fig.add_trace(go.Scatter(
    x=growth_df.index,
    y=growth_df['affordability_stress'],
    fill='tozeroy',
    fillcolor='rgba(204, 51, 17, 0.2)',
    line=dict(color='#CC3311', width=2),
    name='Affordability Stress Index'
))

# Add baseline reference
fig.add_hline(y=100, line_dash="dash", line_color="gray",
              annotation_text="Baseline (100 = Jan 2015 affordability)")

fig.update_layout(
    title='Housing Affordability Stress Index<br><sup>Higher values = worse␣
 ↪affordability relative to wages</sup>',
    xaxis_title='Date',
    yaxis_title='Stress Index (Baseline = 100)',
    template='plotly_white',
    height=450,
)

fig.show()
```

6. Key Insights

Based on our analysis of real FRED and BLS data, here are the key findings about the housing-wage relationship:

[33]:
```
# ============================================================================
# Key Insights Summary
# ============================================================================

# Calculate key metrics
wage_growth_total = growth_df['avg_hourly_earnings_growth'].iloc[-1]
real_wage_growth = growth_df['real_earnings_growth'].iloc[-1]
inflation_total = growth_df['cpi_growth'].iloc[-1]
mortgage_start = combined_df['mortgage_rate'].iloc[0]
mortgage_end = combined_df['mortgage_rate'].iloc[-1]
mortgage_peak = combined_df['mortgage_rate'].max()
stress_current = growth_df['affordability_stress'].iloc[-1]
stress_peak = growth_df['affordability_stress'].max()

print("=" * 65)
print("  KEY INSIGHTS: Housing-Wage Divergence Analysis")
print("=" * 65)
print(f"\n  Analysis Period: {combined_df.index.min().strftime('%Y-%m')} to␣
 ↪{combined_df.index.max().strftime('%Y-%m')}")
print(f"\n  WAGE TRENDS:")
print(f"    • Nominal Wage Growth: +{wage_growth_total:.1f}%")
print(f"    • Real Wage Growth (Inflation-Adjusted): {real_wage_growth:+.1f}%")
print(f"    • Total Inflation (CPI): +{inflation_total:.1f}%")

print(f"\n  HOUSING MARKET:")
print(f"    • Mortgage Rate (Start): {mortgage_start:.2f}%")
print(f"    • Mortgage Rate (Current): {mortgage_end:.2f}%")
print(f"    • Mortgage Rate (Peak): {mortgage_peak:.2f}%")

print(f"\n  AFFORDABILITY:")
print(f"    • Current Stress Index: {stress_current:.1f} (Baseline = 100)")
print(f"    • Peak Stress Index: {stress_peak:.1f}")
if stress_current > 100:
    print(f"    • Status:  Affordability is WORSE than baseline by␣
 ↪{stress_current - 100:.1f} points")
else:
    print(f"    • Status:  Affordability has IMPROVED by {100 - stress_current:.
 ↪1f} points")

print("\n" + "=" * 65)
print("  POLICY IMPLICATIONS")
print("=" * 65)
print("""
1. WAGE-PRICE SPIRAL: If nominal wage growth trails inflation,
   workers lose purchasing power, reducing affordability further.

2. INTEREST RATE SENSITIVITY: Housing affordability is highly
```

```
   sensitive to mortgage rates - even small changes significantly
   impact monthly payments.

3. SUPPLY CONSTRAINTS: Housing starts data can indicate whether
   new construction is keeping pace with demand.

4. REGIONAL VARIATIONS: This national analysis masks significant
   regional differences (see Professional tier for metro-level data).
""")
```

```
======================================================================
 KEY INSIGHTS: Housing-Wage Divergence Analysis
======================================================================

 Analysis Period: 2016-01 to 2024-12

 WAGE TRENDS:
   • Nominal Wage Growth: +40.6%
   • Real Wage Growth (Inflation-Adjusted): +5.2%
   • Total Inflation (CPI): +33.6%

 HOUSING MARKET:
   • Mortgage Rate (Start): 3.97%
   • Mortgage Rate (Current): 6.69%
   • Mortgage Rate (Peak): 7.76%

 AFFORDABILITY:
   • Current Stress Index: 119.8 (Baseline = 100)
   • Peak Stress Index: 145.0
   • Status:   Affordability is WORSE than baseline by 19.8 points

======================================================================
 POLICY IMPLICATIONS
======================================================================

1. WAGE-PRICE SPIRAL: If nominal wage growth trails inflation,
   workers lose purchasing power, reducing affordability further.

2. INTEREST RATE SENSITIVITY: Housing affordability is highly
   sensitive to mortgage rates - even small changes significantly
   impact monthly payments.

3. SUPPLY CONSTRAINTS: Housing starts data can indicate whether
   new construction is keeping pace with demand.

4. REGIONAL VARIATIONS: This national analysis masks significant
   regional differences (see Professional tier for metro-level data).
```

7. Next Steps

1.2.1 Upgrade to Professional Tier

For metro-level and more granular analysis, upgrade to the **Professional Tier** ($149-599/mo):

```
from krl_data_connectors.professional import (
    ZillowConnector,      # Metro-level home values (ZHVI)
    BLSProfessionalConnector,  # Metro-level wages (OES)
    CensusACSConnector,   # County/tract demographics
)

# Metro-level analysis
zillow = ZillowConnector(license_key="YOUR_KEY")
housing = zillow.get_zhvi(geography="metro")
```

1.2.2 Explore More Notebooks

- **02-gentrification-early-warning.ipynb**: Tract-level displacement risk using Census ACS
- **03-economic-mobility-deserts.ipynb**: Opportunity analysis with causal methods
- **10-urban-resilience-dashboard.ipynb**: Complete multi-source analysis workflow

1.2.3 Additional KRL Suite Components

- **krl_models**: `LocationQuotientModel`, `ShiftShareModel` for regional economic analysis
- **krl_geospatial**: Spatial weights, clustering, and mapping tools
- **krl_policy**: Causal inference (Difference-in-Differences, Synthetic Control, RDD)

8. Data Provenance

All data in this notebook comes from official government sources via KRL Data Connectors.

[36]:
```
# ================================================================================
# Data Provenance Documentation
# ================================================================================

provenance = """
## Data Sources

| Dataset | Source | Series ID | Description |
|---------|--------|-----------|-------------|
| Housing Starts | FRED | HOUST | New Residential Construction |
| Mortgage Rate | FRED | MORTGAGE30US | 30-Year Fixed Rate Mortgage |
| CPI | FRED | CPIAUCSL | Consumer Price Index for All Urban Consumers |
| Avg Hourly Earnings | BLS | CES0500000003 | Average Hourly Earnings of All␣
 ↪Employees |
| Unemployment Rate | BLS | LNS14000000 | National Unemployment Rate |

## Access Method
```

```
- **Connector Package**: `krl_data_connectors` v1.0.0
- **Tier**: Community (Free)
- **API Keys Required**: None
- **Rate Limits**: Standard public API limits

## Data Quality Notes

1. **FRED Data**: Updated daily/weekly depending on series
2. **BLS Data**: Monthly releases, typically first Friday of month
3. **Geographic Coverage**: National-level only (Community tier)
4. **Historical Range**: Up to 10 years (Community tier limit)

## Reproducibility

To reproduce this analysis:
```python
from krl_data_connectors.community import FREDBasicConnector, BLSBasicConnector

fred = FREDBasicConnector()
bls = BLSBasicConnector()

housing_df = fred.get_series("HOUST", start_date="2015-01-01")
earnings_df = bls.get_series("CES0500000003")
```
"""

from IPython.display import Markdown
Markdown(provenance)
```

[36]:

1.3 Data Sources

Dataset	Source	Series ID	Description
Housing Starts	FRED	HOUST	New Residential Construction
Mortgage Rate	FRED	MORTGAGE30US	30-Year Fixed Rate Mortgage
CPI	FRED	CPIAUCSL	Consumer Price Index for All Urban Consumers
Avg Hourly Earnings	BLS	CES0500000003	Average Hourly Earnings of All Employees
Unemployment Rate	BLS	LNS14000000	National Unemployment Rate

1.4 Access Method

- **Connector Package**: `krl_data_connectors` v1.0.0
- **Tier**: Community (Free)

- **API Keys Required**: None
- **Rate Limits**: Standard public API limits

1.5 Data Quality Notes

1. **FRED Data**: Updated daily/weekly depending on series
2. **BLS Data**: Monthly releases, typically first Friday of month
3. **Geographic Coverage**: National-level only (Community tier)
4. **Historical Range**: Up to 10 years (Community tier limit)

1.6 Reproducibility

To reproduce this analysis:

```
from krl_data_connectors.community import FREDBasicConnector, BLSBasicConnector

fred = FREDBasicConnector()
bls = BLSBasicConnector()

housing_df = fred.get_series("HOUST", start_date="2015-01-01")
earnings_df = bls.get_series("CES0500000003")
```

[37]:
```
# ============================================================================
# Session Information for Reproducibility
# ============================================================================

import sys

print(" Session Information")
print("=" * 50)
print(f"Python Version: {sys.version}")
print(f"Pandas Version: {pd.__version__}")
print(f"NumPy Version: {np.__version__}")
print()
print(" KRL Suite Packages Used:")
print("   • krl_data_connectors (Community Tier)")
print("   • krl_core (Logging)")
print()
print(f" Execution Completed: {datetime.now().isoformat()}")
```

```
 Session Information
==================================================
Python Version: 3.13.7 (main, Aug 14 2025, 11:12:11) [Clang 17.0.0
(clang-1700.0.13.3)]
Pandas Version: 2.3.3
NumPy Version: 2.3.4

 KRL Suite Packages Used:
   • krl_data_connectors (Community Tier)
```

- `krl_core (Logging)`

```
Execution Completed: 2025-11-27T11:48:45.488172
```

1.7 About the KRL Suite

The **KRL Suite** is a comprehensive socioeconomic analysis platform:

Package	Description	Tier
`krl-data-connectors`	67+ economic data connectors	Community/Pro/Enterprise
`krl-model-zoo`	Regional & forecasting models	Community/Pro
`krl-geospatial-tools`	Spatial analysis & mapping	Community/Pro
`krl-causal-policy-toolkit`	Causal inference methods	Pro/Enterprise
`krl-open-core`	Shared utilities & logging	All tiers

Learn More: github.com/KR-Labs

This notebook is part of the Khipu Socioeconomic Analysis Suite public showcase.
