Question 3

This script demonstrates heterodyning as part of a polyphase interpolator.

In [ ]:
```
from pathlib import Path

import numpy as np
import scipy.fft as fft
import scipy.signal as signal

import matplotlib.pyplot as plt
import seaborn as sns

from a3_config import A3_ROOT, SAVEFIG_CONFIG
```

Polyphase Upsample & Heterodyne

In [ ]:
```
# Import polyphase downsampled signal from Question 1
t_signal, x_signal = np.load(Path(A3_ROOT, "output", "q1_signal_out.npy"))

# Import Kaiser LPF from Question 1
x_kaiser_lpf = np.load(Path(A3_ROOT, "output", "q1_kaiser_lpf.npy"))
```

In [ ]:
```
N = len(x_kaiser_lpf)   # filter length
L = 80                  # upsampling rate, equal to M from Question 1
FS = 0.5                # sampling frequency, kHz
F_CARRIER = 10          # frequency shift, kHz

# Reshape filter coefficients into matrix, zero padded to multiple of L
Z = L - (N % L)
polyfilt = np.concatenate([x_kaiser_lpf, np.zeros(Z)])
polyfilt = polyfilt.reshape(int((N + Z) / L), L).T  # reshape row-major then T
# Apply heterodyning (frequency shifting) w/ 10 kHz carrier
k = F_CARRIER / FS

for i in range(L):
    polyfilt[i] *= np.cos(2 * np.pi * i * k / L)
```

In [ ]:
```
# Concatenate results into output array, which becomes the filtered signal
x_polyfilt = []
for i in range(L):
    x_polyfilt.append(signal.convolve(polyfilt[i], x_signal))
x_polyfilt = np.array(x_polyfilt).flatten("F")

# As before, remove transient edge effects
x_polyfilt = x_polyfilt[(N+Z-L)//2:-(N+Z-L)//2]

# Calculate transform for plotting
h_polyfilt = fft.fft(x_polyfilt, 8192)[:4096]

# Construct time and frequency axes for plotting
t_polyfilt = np.arange(0, 50, 50 / len(x_polyfilt))
f_polyfilt = fft.fftfreq(8192, 50 / len(x_polyfilt))[:4096]
```

```python
# Plot the polyphase downsampled signal
fig, axs = plt.subplots(1, 2, figsize=(7.5, 1.5))

sns.lineplot(x=t_polyfilt, y=x_polyfilt.real, ax=axs[0], lw=1)
sns.lineplot(x=f_polyfilt, y=np.abs(h_polyfilt), ax=axs[1], lw=1)

axs[0].set_xlabel("Time (ms)")
axs[1].set_xlabel("Frequency (kHz)")
axs[1].set_xlim([8.685, 11.315])

fig.tight_layout()
fig.savefig(Path(A3_ROOT, "output", "q3_heterodyne.png"), **SAVEFIG_CONFIG)
```

Performance Comparison

In [ ]:

```python
import time
from tqdm import trange

N_TRIALS = 10000
msfmt = lambda t: f'{(t * 1000 / N_TRIALS):.5f}'

time_start = time.time()
for _ in trange(N_TRIALS):
    x_polyfilt = []
    for i in range(L):
        x_polyfilt.append(signal.convolve(polyfilt[i], x_signal))
    x_polyfilt = np.array(x_polyfilt).flatten("F")
time_elapsed = time.time() - time_start
print(f"Polyphase interpolator w/ heterodyning ({N_TRIALS} trials): "
      f"{msfmt(time_elapsed)} ms")
```