6.4 Problems

3A

Problem 6.1 STANISLAW Explain, in your own words (supplementary drawings are ok), what the *frequency domain* is, how we derive models in it, and why it is useful.

Problem 6.2 PUG Consider the function

$$f(t) = 8\cos(t) + 6\sin(2t) + \sqrt{5}\cos(4t) + 2\sin(4t) + \cos(6t - \pi/2).$$

(a) Find the (harmonic) magnitude and (harmonic) phase of its Fourier series components. (b) Sketch its magnitude and phase spectra. *Hint: no Fourier integrals are necessary to solve this problem.*

Problem 6.3 PONYO Consider the function with $a > 0$

$$f(t) = e^{-a|t|}.$$

From the transform definition, derive the Fourier transform $F(\omega)$ of $f(t)$. Simplify the result such that it is clear the expression is real (no imaginary component).

Problem 6.4 SEESAW Consider the periodic function $f : \mathbb{R} \to \mathbb{R}$ with period T defined for one period as

$$f(t) = at \quad \text{for } t \in (-T/2, T/2] \tag{6.9}$$

where $a, T \in \mathbb{R}$. Perform a fourier series analysis on f. Letting $a = 5$ and $T = 1$, plot f along with the partial sum of the fourier series synthesis, the first 50 nonzero components, over $t \in [-T, T]$.

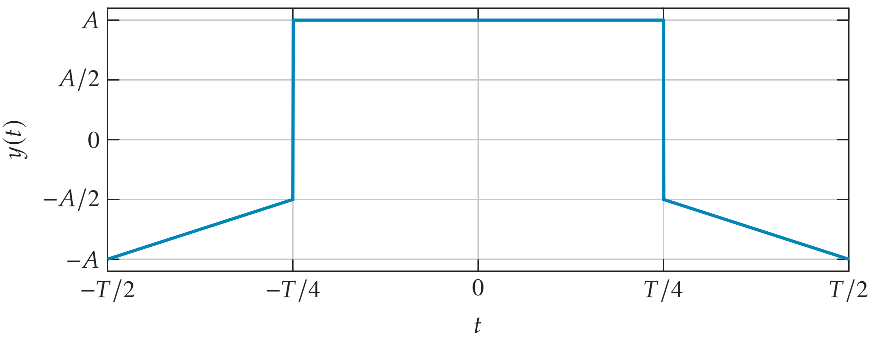

Figure 6.9. one period T of the function $y(t)$. Every line that appears straight is so.

Problem 6.5 TOTORO Consider a periodic function $y(t)$ with some period $T \in \mathbb{R}$ and some parameter $A \in \mathbb{R}$ for which one period is shown in figure 6.9.

a. Perform a *trigonometric* Fourier series analysis of $y(t)$ and write the Fourier series $Y(\omega)$.
b. Plot the harmonic amplitude spectrum of $Y(\omega)$ for $A = T = 1$. Consider using computing software.
c. Plot the phase spectrum of $Y(\omega)$ for $A = T = 1$. Consider using computing software.

Problem 6.6 MALL Consider the function $f : \mathbb{R} \to \mathbb{R}$ defined as

$$f(t) = \begin{cases} a - a|t|/T & \text{for } t \in [-T, T] \\ 0 & \text{otherwise} \end{cases} \tag{6.10}$$

where $a, T \in \mathbb{R}$. Perform a fourier transform analysis on f, resulting in $F(\omega)$. Plot F for various a and T.

Problem 6.7 MIYAZAKI Consider the function $f : \mathbb{R} \to \mathbb{R}$ defined as

$$f(t) = ae^{-b|t-T|} \tag{6.11}$$

where $a, b, T \in \mathbb{R}$. Perform a fourier transform analysis on f, resulting in $F(\omega)$. Plot F for various a, b, and T.

Problem 6.8 HAKU Consider the function $f : \mathbb{R} \to \mathbb{R}$ defined as

$$f(t) = a \cos \omega_0 t + b \sin \omega_0 t \tag{6.12}$$

where $a, b, \omega_0 \in \mathbb{R}$ constants. Perform a fourier transform analysis on f, resulting in $F(\omega)$.[4]

Problem 6.9 SECRETS This exercise encodes a "secret word" into a sampled waveform for decoding via a *discrete fourier transform* (DFT). The nominal goal of the exercise is to decode the secret word. Along the way, plotting and interpreting the DFT will be important.

First, load relevant packages.

```
import numpy as np
import matplotlib.pyplot as plt
from IPython.display import display, Markdown, Latex
```

We define two functions: `letter_to_number` to convert a letter into an `integer` index of the alphabet (a becomes 1, b becomes 2, etc.) and `string_to_number_list` to convert a string to a `list` of `int`s, as shown in the example at the end.

```
def letter_to_number(letter):
    return ord(letter) - 96

def string_to_number_list(string):
    out = [] # list
    for i in range(0,len(string)):
        out.append(letter_to_number(string[i]))
    return out # list

print(f"aces = { string_to_number_list('aces') }")
```

```
aces = [1, 3, 5, 19]
```

Now, we encode a code string `code` into a signal by beginning with "white noise," which is *broadband* (appears throughout the spectrum) and adding to it `sin` functions with amplitudes corresponding to the letter assignments of the code and harmonic corresponding to the position of the letter in the string. For instance, the string `'bad'` would be represented by noise plus the signal

$$2\sin 2\pi t + 1\sin 4\pi t + 4\sin 6\pi t. \tag{6.13}$$

Let's set this up for secret word `'chupcabra'`.

```
N = 2000
Tm = 30
```

4. It may be alarming to see a Fourier transform of a periodic function! Strictly speaking, it does not exist; however, if we extend the transform to include the *distribution* (not actually a function) Dirac $\delta(\omega)$, the modified-transform does exist and is given in table B.2.

4. Python code in this section was generated from a Jupyter notebook named `random_signal_fft.ipynb` with a `python3` kernel.

```
T = float(Tm)/float(N)
fs = 1/T
x = np.linspace(0, Tm, N)
noise = 4*np.random.normal(0, 1, N)
code = 'chupcabra' # the secret word
code_number_array = np.array(string_to_number_list(code))
y = np.array(noise)
for i in range(0,len(code)):
    y = y + code_number_array[i]*np.sin(2.*np.pi*(i+1.)*x)
```

For proper decoding, later, it is important to know the fundamental frequency of the generated data.

```
print(f"fundamental frequency = {fs} Hz")
```

```
fundamental frequency = 66.66666666666667 Hz
```

Now, we plot.

```
fig, ax = plt.subplots()
plt.plot(x,y)
plt.xlim([0,Tm/4])
plt.xlabel('time (s)')
plt.ylabel('$y_n$')
plt.show()
```

Finally, we can save our data to a numpy file `secrets.npy` to distribute our message.

```
np.save('secrets',y)
```

Now, I have done this (for a different secret word!) and saved the data; download it here:

https://math.ricopic.one/sg

SG

In order to load the `.npy` file into *Python*, we can use the following command.

```
secret_array = np.load('secrets.npy')
```

Your job is to (a) perform a DFT, (b) plot the spectrum, and (c) decode the message! Here are a few hints.

1. Use `from scipy import fft` to do the DFT.
2. Use a `hanning` window to minimize the end-effects. See `numpy.hanning` for instance. The `fft` call might then look like

   ```
   2*fft(np.hanning(N)*secret_array)/N
   ```

   where `N = len(secret_array)`.

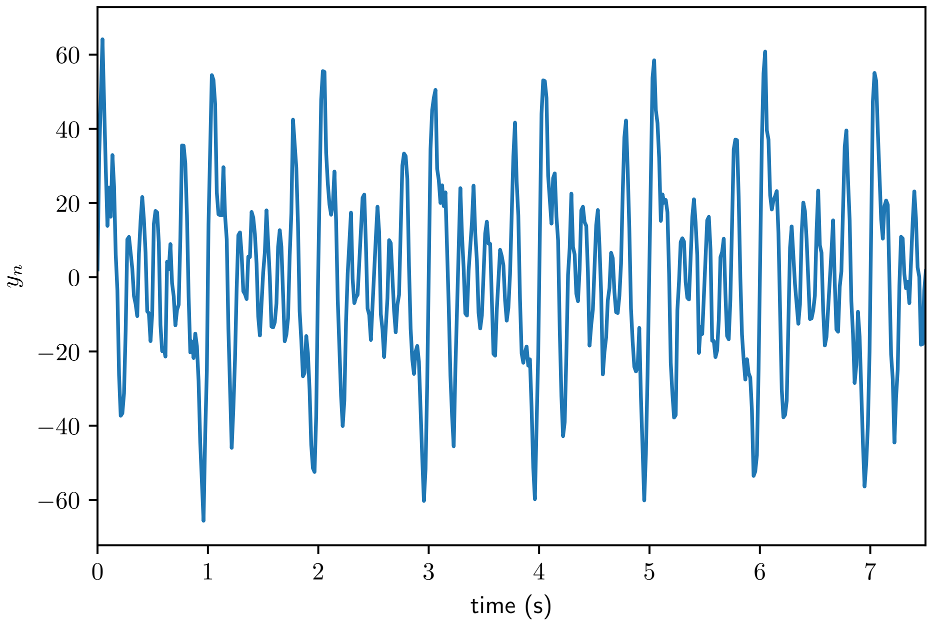

Figure 6.10. the chupacabra signal.

3. Use only the *positive* spectrum; you can lop off the negative side and double the positive side.

Problem 6.10 SOCIETY Derive a fourier transform property for expressions including function $f : \mathbb{R} \rightarrow \mathbb{R}$ for

$$f(t)\cos(\omega_0 t + \psi)$$

where $\omega_0, \psi \in \mathbb{R}$.

Problem 6.11 FLAPPER Consider the function $f : \mathbb{R} \rightarrow \mathbb{R}$ defined as

$$f(t) = a u_s(t) e^{-bt} \cos(\omega_0 t + \psi) \tag{6.14}$$

where $a, b, \omega_0, \psi \in \mathbb{R}$ and $u_s(t)$ is the unit step function. Perform a fourier transform analysis on f, resulting in $F(\omega)$. Plot F for various a, b, ω_0, ψ and T.

Problem 6.12 EASTEGG Consider the function $f : \mathbb{R} \rightarrow \mathbb{R}$ defined as

$$f(t) = g(t)\cos(\omega_0 t) \tag{6.15}$$

where $\omega_0 \in \mathbb{R}$ and $g : \mathbb{R} \to \mathbb{R}$ will be defined in each part below. Perform a fourier transform analysis on f for each g below for $\omega_1 \in \mathbb{R}$ a constant and consider how things change if $\omega_1 \to \omega_0$.

a. $g(t) = \cos(\omega_1 t)$
b. $g(t) = \sin(\omega_1 t)$

Problem 6.13 SAVAGE An instrument called a "lock-in amplifier" can measure a sinusoidal signal $A\cos(\omega_0 t + \psi) = a\cos(\omega_0 t) + b\sin(\omega_0 t)$ at a known frequency ω_0 with exceptional accuracy even in the presence of significant noise $N(t)$. The workings of these devices can be described in two operations: first, the following operations on the signal with its noise, $f_1(t) = a\cos(\omega_0 t) + b\sin(\omega_0 t) + N(t)$,

$$f_2(t) = f_1(t)\cos(\omega_1 t) \quad \text{and} \quad f_3(t) = f_1(t)\sin(\omega_1 t). \tag{6.16}$$

where $\omega_0, \omega_1, a, b \in \mathbb{R}$. Note the relation of this operation to the Fourier transform analysis of problem 6.12. The key is to know with some accuracty ω_0 such that the instrument can set $\omega_1 \approx \omega_0$. The second operation on the signal is an aggressive low-pass filter. The filtered f_2 and f_3 are called the *in-phase* and *quadrature* components of the signal and are typically given a complex representation

$$(\text{in-phase}) + j\,(\text{quadrature}).$$

Explain with fourier transform analyses on f_2 and f_3

a. what $F_2 = \mathcal{F}(f_2)$ looks like,
b. what $F_3 = \mathcal{F}(f_3)$ looks like,
c. why we want $\omega_1 \approx \omega_0$,
d. why a low-pass filter is desirable, and
e. what the time-domain signal will look like.

Problem 6.14 STRAWMAN Consider again the lock-in amplifier explored in problem 6.13. Investigate the lock-in amplifier numerically with the following steps.

a. Generate a noisy sinusoidal signal at some frequency ω_0. Include enough broadband white noise that the signal is invisible in a time-domain plot.
b. Generate f_2 and f_3, as described in problem 6.13.
c. Apply a time-domain discrete low-pass filter to each $f_2 \mapsto \phi_2$ and $f_3 \mapsto \phi_3$, such as scipy's `scipy.signal.sosfiltfilt`, to complete the lock-in amplifier operation. Plot the results in time and as a complex (polar) plot.
d. Perform a discrete fourier transform on each $f_2 \mapsto F_2$ and $f_3 \mapsto F_3$. Plot the spectra.
e. Construct a frequency domain low-pass filter F and apply it (multiply!) to each $F_2 \mapsto F_2'$ and $F_3 \mapsto F_3'$. Plot the filtered spectra.
f. Perform an inverse discrete fourier transform to each $F_2' \mapsto f_2'$ and $F_3' \mapsto f_3'$. Plot the results in time and as a complex (polar) plot.

g. Compare the two methods used, i.e. time-domain filtering versus frequency-domain filtering.

7 Partial Differential Equations

UA

An **ordinary differential equation** is one with (ordinary) derivatives of functions of a single variable each—time, in many applications. These typically describe quantities in some sort of **lumped-parameter** way: mass as a "point particle," a spring's force as a function of time-varying displacement across it, a resistor's current as a function of time-varying voltage across it. Given the simplicity of such models in comparison to the wildness of nature, it is quite surprising how well they work for a great many phenomena. For instance, electronics, rigid body mechanics, population dynamics, bulk fluid mechanics, and bulk heat transfer can be lumped-parameter modeled.

However, as we saw in (**vecs**), there are many phenomena of which we require more detailed models. These include:

- detailed fluid mechanics,
- detailed heat transfer,
- solid mechanics,
- electromagnetism, and
- quantum mechanics.

In many cases, what is required to account for is the **time-varying spatial distribution** of a quantity. In fluid mechanics, we treat a fluid as having quantities such as density and velocity that vary continuously over space and time. Deriving the governing equations for such phenomena typically involves vector calculus; we observed in (**vecs**) that statements about quantities like the divergence (e.g., continuity) can be made about certain scalar and vector fields. Such statements are governing equations (e.g., the continuity equation) and they are **partial differential equations** (PDEs) because the quantities of interest, called **dependent variables** (e.g., density and velocity), are both temporally and spatially varying (temporal and spatial variables are therefore called **independent variables**).

In this chapter, we explore the **analytic solution** of PDEs. This is related to but distinct from the **numeric solution** (i.e., simulation) of PDEs, which is another important topic. Many PDEs have no known analytic solution, so for these numeric solution is the best available option.[1] However, it is important to note that the insight one can gain from an analytic solution is often much greater than that from a numeric solution. This is easily understood when one considers that a numeric solution is an approximation for a specific set of initial and boundary conditions. Typically, very little can be said of what would happen in general, although this is often what we seek to know. So, despite the importance of numeric solution, one should always prefer an analytic solution.

Three good texts on PDEs for further study are (Kreyszig 2011; Ch. 12), (Strauss 2007), and (Haberman 2018).

7.1 Classifying PDEs

VV

PDEs often have an infinite number of solutions; however, when applying them to physical systems, we usually assume that a deterministic, or at least a probabilistic, sequence of events will occur. Therefore, we impose additonal constraints on a PDE, usually in the form of

1. **initial conditions**, values of independent variables over all space at an initial time and
2. **boundary conditions**, values of independent variables (or their derivatives) over all time.

Ideally, imposing such conditions leaves us with a **well-posed problem**, which has three aspects. (Bove, Colombini, and Santo 2006; § 1.5)

existence There exists at least one solution.
uniqueness There exists at most one solution.
stability If the PDE, boundary conditons, or initial conditions are changed slightly, the solution changes only slightly.

As with ODEs, PDEs can be **linear** or **nonlinear**; that is, the dependent variables and their derivatives can appear in only linear combinations (linear PDE) or in one or more nonlinear combination (nonlinear PDE). As with ODEs, there are more known analytic solutions to linear PDEs than nonlinear PDEs.

The **order** of a PDE is the order of its highest partial derivative. A great many physical models can be described by **second-order PDEs** or systems thereof. Let u be an independent scalar variable, a function of m temporal and spatial variables $x_i \in \mathbb{R}^n$. A second-order linear PDE has the form, for coefficients $\alpha, \beta, \gamma, and \delta$, and

1. There are some analytic techniques for gaining insight into PDEs for which there are no known solutions, such as considering the *phase space*. This is an active area of research; for more, see (Bove, Colombini, and Santo 2006).