Τηλεπικοινωνιακά Συστήματα

με Python

Θωμάς Καμαλάκης

Σημειώσεις του μαθήματος Τηλεπικοινωνιακά Συστήματα

Τμήμα Πληροφορικής και Τηλεματικής, Χαροκόπειο Πανεπιστήμιο Αθηνών
Πρώτη έκδοση, Ιανουάριος 2022

1. Εισαγωγή

Τα τηλεπικοινωνιακά συστήματα ή όπως είναι ο πιο μοντέρνος τίτλος τους "συστήματα επικοινωνιών" είναι παντού γύρω μας. Καθορίζουν σε μεγάλο βαθμό την ανθρώπινη δραστηριότητα. Το 2021, το μέγεθος της αγοράς των επικοινωνιών αποτιμήθηκε στα 2.5 τρισεκατομμύρια δολάρια και περιλαμβάνει τους παρόχους σταθερής και κινητής πρόσβασης, της εταιρείες που υλοποιούν τα συστήματα που χρησιμοποιούνται στα διάφορα δίκτυα και τους μεταπωλητές.

Στο μάθημα αυτό θα εξετάσουμε τις βασικές αρχές που διέπουν τα τηλεπικοινωνιακά συστήματα. Θα δούμε πως αποτυπώνεται η πληροφορία στον πομπό, πως μεταδίδεται μέσω του καναλιού και πως ο δέκτης επιχειρεί να ξεκαθαρίσει το σήμα που λαμβάνει από τον θόρυβο και να αποφασίσει ποιο ήταν το μήνυμα που ήθελε να στείλει ο πομπός. Προσπαθούμε να ξεφύγουμε από την παραδοσιακή προσέγγιση της διδασκαλίας του μαθήματος αυτού η οποία συνίσταται στην καθαρά μαθηματική-θεωρητική ανάλυση των συστημάτων και να δούμε τα συστήματα αυτά σε μεγαλύτερη λεπτομέρεια χρησιμοποιώντας την `Python`. Κάθετι θεωρητικό που θα μαθαίνουμε, θα επιχειρούμε να το δούμε και στον υπολογιστή με την ελπίδα ότι θα το καταλάβουμε καλύτερα.

1.1 Τι θα μάθουμε

Σε ότι αφορά το θεωρητικό μέρος, το μάθημα στοχεύει να σας μάθει τα διάφορα σχήματα διαμόρφωσης που χρησιμοποιούνται σε πρακτικές εφαρμογές όπως η διαμόρφωση πλάτους, φάσης κτλ. Θα καλύψουμε και θέματα όπως η φύση και η κωδικοποίηση της πηγής της πληροφορίας αλλά και η κωδικοποίηση για την προστασία από σφάλματα (forward error correction - FEC). Σε ότι αφορά το πρακτικό μέρος, θα πρέπει να προσφέρουμε και μία συνεκτική εισαγωγή στην `Python` δίνοντας έμφαση σε βασικές βιβλιοθήκες που χρησιμοποιούνται για την προσομοίωση συστημάτων όπως η `numpy` και η `scipy` αλλά και την απεικόνιση των αποτελεσμάτων όπως η `matplotlib`. Τέλος θα χρειαστεί να καλύψουμε και ορισμένα θέματα που άπτονται της προσομοίωσης συστημάτων όπως για παράδειγμα η προσομοίωση Monte Carlo αλλά και ορισμένες βασικές μεθόδους αριθμητικής ανάλυσης όπως η αριθμητική

ολοκλήρωση.

1.2 Τι δεν θα μάθουμε

Αν και από ότι βλέπετε η Python είναι ο βασικός τρόπος που θα δοκιμάζουμε πράγματα αλλά οι σημειώσεις αυτές δεν αποσκοπούν να σας διδάξουν Python, δεν πρόκειται δηλαδή ένα μάθημα προγραμματισμού. Ωστόσο θα δούμε μερικές βασικές αρχές της γλώσσας καθώς και τις βιβλιοθήκες που αναφέραμε και παραπάνω. Ελπίζω ότι θα αποκομίσετε γνώσεις που θα σας βοηθήσουν να μάθετε πιο εύκολα την Python και να την εφαρμόσετε και σε διάφορα άλλα πεδία όπως η ανάλυση δεδομένων, η τεχνητή νοημοσύνη ή ακόμα και το web development.

1.3 Προαπαιτούμενα

Είναι καλό να έχετε περάσει ή έστω να έχετε παρακολουθήσει ένα μάθημα που να έχει να κάνει με Σήματα και Συστήματα καθώς και ένα μάθημα Στατιστικής και Πιθανοτήτων. Θα προσπαθήσουμε να αναλύσουμε όλες τις απαραίτητες έννοιες στα μάθημα αυτό ωστόσο αυτό μπορεί να γίνει ως ένα βαθμό.

1.4 Πηγές

Οι σημειώσεις αυτές έρχονται να συμπληρώσουν το ήδη υπάρχον υλικό που υπάρχει για το μάθημα στο διαδίκτυο. Καταρχήν, οι διαλέξεις του μαθήματος για τα ακαδημαϊκά έτη 2020-2021 και 2021-2022 υπάρχουν στο youtube, στο κανάλι του Τμήματος Πληροφορικής και Τηλεματικής[1] η απευθείας στο κανάλι του διδάσκοντα[2]. Οι διαφάνειες του μαθήματος είναι ανεβάσμενες στο e-class του Πανεπιστημίου[3] (δεν χρειάζεστε να είστε εγγεγραμμένοι στο σύστημα για να έχετε πρόσβαση). Σε ότι αφορά τα βιβλία που προτείνονται για το μάθημα, η αλήθεια είναι πως προσωπικά δεν μπορώ να σκεφτώ κάποιο βιβλίο που να καλύπτει πλήρως το αντικείμενο. Υπάρχουν εξαιρετικά βιβλία για το θεωρητικό υπόβαθρο των τηλεπικοινωνιακών συστημάτων όπως των Proakis/Salehi[4] καθώς και των Καραγιαννίδη/Πάππη[5]. Οι παραπάνω τίτλοι που προτείνονται και στο ηλεκτρονικό σύστημα του Ευδόξου ωστόσο δεν καλύπτουν το μέρος του μαθήματος που έχει να κάνει με την Python και προσωπικά ελπίζω το κενό αυτό να καλυφθεί με τις παρούσες σημειώσεις.

1.5 Η γλώσσα Python

Η γλώσσα Python είναι μία γλώσσα υψηλού επιπέδου και γενικού σκοπού, δηλαδή σας γλιτώνει από το να γράφετε κώδικα μηχανής (!) και μπορείτε να την χρησιμοποιήσετε για οποιοδήποτε σκοπό ή σχεδόν οποιοδήποτε - μην πάτε να γράψετε λειτουργικό σύστημα σε Python!

[1] https://www.youtube.com/channel/UCEHkYirpXF1nSLxDCrfDZ4A
[2] https://www.youtube.com/channel/UCCcFlL-4fM9uhAJkJtMMwOw
[3] https://eclass.hua.gr/courses/DIT149/
[4] Συστήματα Τηλεπικοινωνιών (2015), εκδόσεις Φούντας.
[5] Τηλεπικοινωνιακά Συστήματα, 4η Έκδοση, εκδόσεις Τζιόλα.

1.6 Περιβάλλοντα Ανάπτυξης

Υπάρχει μία πληθώρα περιβαλλόντων ανάπτυξης για την γλώσσα Python όπως το PyCharm[6] του οποίου η βασική έκδοση είναι δωρεάν. Αν είστε διατεθειμένοι να εγκαταστήσετε το Anaconda[7] τότε μπορείτε να χρησιμοποιήσετε και το JupyterLab[8]. Προσωπικά χρησιμοποιώ για το μάθημα το Spyder[9], επειδή επί πολλά χρόνια χρησιμοποιούσα το MATLAB και το Spyder ίσως είναι το πλησιέστερο σε αυτό. Φυσικά μπορείτε να χρησιμοποιήσετε οποιοδήποτε editor της αρεσκείας σας όπως το Atom[10] σε συνδυασμό με ένα Python console (ιδιαίτερα χρήσιμη είναι η ipython[11]).

1.7 Βιβλιοθήκες

Η Python διαθέτει πάρα πολλές βιβλιοθήκες και μάλιστα ορισμένες είναι εστιασμένες στα συστήματα επικοινωνιών όπως η Komm[12]. Για λόγους εκπαιδευτικούς όμως, πήρα την απόφαση να ξεκινήσουμε από πολύ βασικές βιβλιοθήκες αριθμητικής ανάλυσης όπως η numpy και η scipy και να φτιάξουμε σταδιακά μία δική μας βιβλιοθήκη για προσομοίωση συστημάτων. Επίσης θα χρησιμοποιήσουμε όσο μπορούμε την matplotlib για την απεικόνιση των αποτελεσμάτων μας. Πρόκειται για αρκετά διαδεδομένες και καλά συντηρημένες βιβλιοθήκες που χρησιμοποιούνται και πολλές άλλες εφαρμογές.

1.8 Επικοινωνία

Θα χαρώ πολύ να μάθω τα σχόλια σας για τις σημειώσεις. Αν δεν είστε φοιτητές του τμήματος μας μπορείτε να με βρείτε στο e-mail thkam at hua dot gr. Με συγχωρείτε που το γράφω έτσι αλλά διαφορετικά θα γέμιζα από ανεπιθύμητα μηνύματα από crawlers/bots.

[6] https://www.jetbrains.com/pycharm/
[7] https://www.anaconda.com/
[8] https://jupyter.org/
[9] https://www.spyder-ide.org/
[10] https://atom.io/
[11] https://ipython.org/
[12] https://komm.readthedocs.io/

2. Hello World!

2.1 Εισαγωγή

Στο παρόν κεφάλαιο θα δώσουμε μερικές βασικές οδηγίες πως να εγκαταστήσετε το απαραίτητο λογισμικό στον υπολογιστή σας. Στη συνέχεια θα τρέξουμε μερικά βασικά παραδείγματα για να βεβαιωθούμε ότι έχουμε εγκαταστήσει ότι χρειαζόμαστε.

2.2 Λειτουργικό Σύστημα

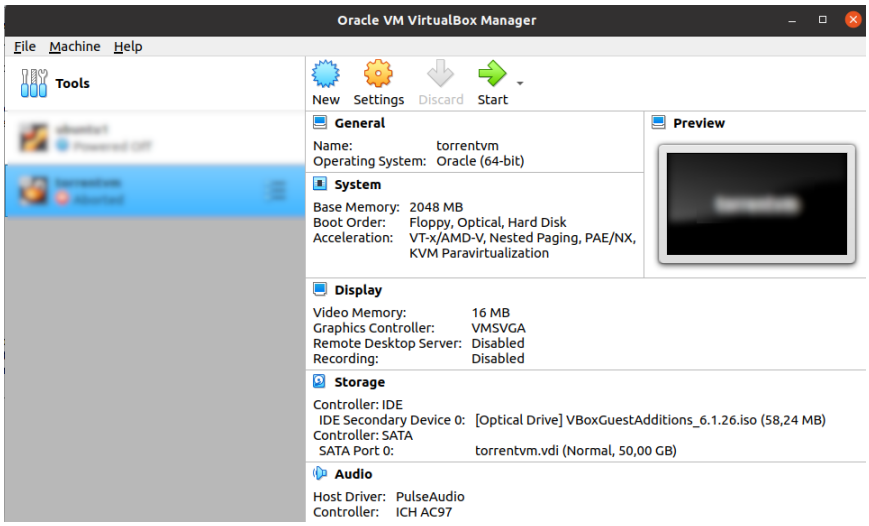
Όπως μπορεί και να γνωρίζετε είμαι μεγάλος οπαδός του `Linux` και επομένως έχω γράψει αυτές τις σημειώσεις θεωρώντας ότι θα το χρησιμοποιήσετε και εσείς. Το πιο πιθανό όμως είναι να έχετε υπολογιστή με `Windows` ή κάποιο `Mac`. Στην περίπτωση αυτή προσωπικά θα συνιστούσα να χρησιμοποιήσετε το `Virtualbox`[1] το οποίο θα σας επιτρέψει να φτιάξετε μία μικρή ιδεατή μηχανή Linux που θα χρησιμοποιήσουμε για τις ανάγκες του μαθήματος. Φυσικά τίποτα δεν σας σταματά να βρείτε ένα παλιότερο υπολογιστή και να εγκαταστήσετε εκεί το `Linux`.[2]. Η έκδοση του Virtualbox που χρησιμοποιώ εδώ είναι η 6.1 αλλά φαντάζομαι ότι ο τρόπος δημιουργίας της ιδεατής μηχανής θα είναι παρόμοιος σε παλιότερες αλλά και σε νεότερες εκδόσεις του. Για την εγκατάσταση του Virtualbox θα πρέπει να συμβουλευτείτε την αντίστοιχη ιστοσελίδα και είναι σχετικά απλή διαδικασία.

2.3 Δημιουργία της ιδεατής μηχανής

Εφόσον έχετε εγκαταστήσει το Virtualbox μπορείτε να το ξεκινήσετε και να επιλέξετε από το μενού, Machine > New

[1] `https://www.virtualbox.org/`

[2] Προσωπικά χρησιμοποιώ το Linux πολλά χρόνια στον βασικό μου υπολογιστή και έχω να ασχοληθώ με άλλα λειτουργικά συστήματα πολλά χρόνια. Αλλά μάλλον αποτελώ την εξαίρεση!


Oracle VM VirtualBox Manager
File Machine Help
Tools
New Settings Discard Start
General
Name: torrentvm
Operating System: Oracle (64-bit)
System
Base Memory: 2048 MB
Boot Order: Floppy, Optical, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging, PAE/NX, KVM Paravirtualization
Preview
Display
Video Memory: 16 MB
Graphics Controller: VMSVGA
Remote Desktop Server: Disabled
Recording: Disabled
Storage
Controller: IDE
IDE Secondary Device 0: [Optical Drive] VBoxGuestAdditions_6.1.26.iso (58,24 MB)
Controller: SATA
SATA Port 0: torrentvm.vdi (Normal, 50,00 GB)
Audio
Host Driver: PulseAudio
Controller: ICH AC97

3. Σήματα

3.1 Εισαγωγή

Τώρα που έχουμε εγκαταστήσει την Python και τις βιβλιοθήκες τις είμαστε έτοιμοι να ξεκινήσουμε το ταξίδι μας στο χώρο των τηλεπικοινωνιών. Τα *σήματα* είναι πολύ βασική έννοια στον χώρο των τηλεπικοινωνιών καθώς μεταφέρουν την πληροφορία από τον πομπό στον δέκτη. Στο παρόν κεφάλαιο θα ασχοληθούμε με τα σήματα και θα χρησιμοποιήσουμε την Python για να δούμε κάποια από αυτά στην πράξη.

3.2 Η έννοια του σήματος

Ως *σήμα* ορίζουμε την μεταβολή ενός φυσικού μεγέθους $x = x(t)$, π.χ. τάση, ρεύμα, ισχύς ως προς τον χρόνο t. Στην φύση συναντούμε κατά κύριο λόγο *αναλογικά* σήματα που λαμβάνουν δυνητικά οποιαδήποτε τιμή μέσα σε ένα διάστημα τιμών. Στην εικόνα 3.1 δείχνουμε ένα παράδειγμα αναλογικού σήματος.

Τα αναλογικά σήματα χρησιμοποιούνταν στο παρελθόν στην τηλεφωνία και στην ραδιοφωνία, ωστόσο στις μέρες μας η πλειοψηφία των συστημάτων επικοινωνιών βασίζεται στην μετάδοση *ψηφιακών* σημάτων. Ένα ψηφιακό σήμα $x(t)$ εξ ορισμού μπορεί να πάρει συγκεκριμένες διακριτές τιμές x_i. Η Εικόνα 3.2 δείχνει ένα ψηφιακό σήμα που παίρνει δύο διαφορετικές τιμές x_i, μία υψηλή και μία χαμηλή.

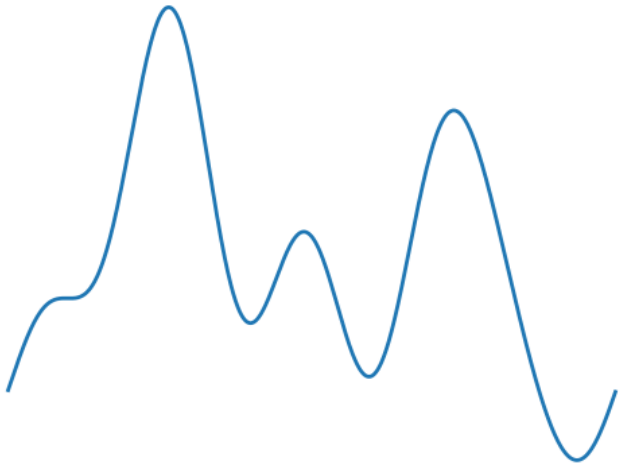
Στις τηλεπικοινωνίες συναντούμε συχνά δύο παραδείγματα σημάτων: τον τετραγωνικό παλμό και το φέρον σήμα. Ο *τετραγωνικός παλμός* δίνεται από την σχέση:

$$p(t) = \begin{cases} 1 & , -\frac{T_1}{2} \leq t \leq \frac{T_1}{2} \\ 0 & , \text{διαφορετικά} \end{cases} \tag{3.1}$$

Πρόκειται δηλαδή για ένα σήμα που είναι παντού μηδέν εκτός μίας διάρκειας T_1 με κέντρο το $t = 0$ όπου είναι ίσο με ένα. Πολλά σήματα συστημάτων επικοινωνιών αποτελούνται από διαδοχικούς τετραγωνικούς παλμούς ή προσεγγίσεις τους.

Ένα άλλο παράδειγμα σήματος είναι το *φέρον* σήμα που δίνεται από την σχέση:

$$x_c(t) = A\cos(2\pi f_0 t + \phi) \tag{3.2}$$



Εικόνα 3.1: Ένα αναλογικό σήμα.

Εικόνα 3.2: Ένα ψηφιακό σήμα.

Πρόκειται δηλαδή για ένα συνημιτονοειδές σήμα με συχνότητα f_0 και αρχική φάση ίση με ϕ. Το παραπάνω σήμα είναι *περιοδικό* καθώς επαναλαμβάνεται κάθε $T = \frac{1}{f_0}$ δηλαδή

$$x_c(t+T) = x_c(t). \tag{3.3}$$

3.3 Σήματα στην Python

Θα χρησιμοποιήσουμε την Python για να δημιουργήσουμε τα δύο σήματα που είδαμε στην προηγούμενη ενότητα, δηλαδή έναν τετραγωνικό παλμό και ένα φέρον. Καταρχήν είναι καλή ιδέα ότι φτιάχνουμε να το βάζουμε σε μία βιβλιοθήκη το οποίο θα καλούμε κάθε φορά στα προγράμματα μας. Δημιουργούμε ένα αρχείο Python το οποίο θα το ονομάσουμε `commlib.py`. Στο listing 3.1 δείχνουμε την πρώτη έκδοση του `commlib.py`. Στην ουσία εδώ ορίζουμε μία κλάση της Python την `signal` η οποία περιέχει τα απαραίτητα χαρακτηριστικά ενός *αναλογικού* σήματος. Η πρώτη γραμμή κάνει import την βιβλιοθήκη `numpy` της Python.

```
import numpy as np
import matplotlib.pyplot as plt

#---level1
def level(t, x, lvl = -3, post_only = False, return_t = False):
    if post_only:
        ipos = np.where(t >= 0)
        x = x[ipos]
        t = t[ipos]

    X = np.abs(x) ** 2.0
    XdB = 10 * np.log10(X)


    imax = np.argmax( XdB )
    Xmax = XdB[imax]
    XdB_lvl = Xmax - lvl

    i = imax
    Xcur = XdB[i]

    while (i < x.size) and ( Xcur >= XdB_lvl ):
        i += 1
        Xcur = XdB[i]
    if (Xcur < XdB_lvl):
        tright = np.interp(XdB_lvl, XdB[imax:(i+1)], t[imax:(i+1)] )
    else:
        tright = np.Inf


    i = imax
    Xcur = XdB[i]

    while (i >= 0) and ( Xcur >= XdB_lvl ):
        i -= 1
        Xcur = XdB[i]

    if (Xcur < XdB_lvl):
        tleft = np.interp(XdB_lvl, XdB[i:imax], t[i:imax] )
    else:
```

```
        tleft = -np.Inf

    if return_t:
        return {'tleft' : tleft,
                'tright' : tright,
                'B' : tright - tleft}
    else:
        return tright - tleft
#---level2

#---signal1
class signal:
    """
    The basic signal class
    """
    def __init__(self, t = None, samples = None):
        """
        Initialize the signal class
        """
        self.t = t
        if self.t is not None:
            self.Dt = t[1] - t[0]
            self.N = t.size
        else:
            self.N = 0
            self.Dt = None

        self.samples = samples

    def plot(self, line_type = '-o', label = None):
        """
        Plot the signal
        """
        plt.plot(self.t, self.samples, label = label)
        plt.xlabel('t')
        plt.ylabel('x')

    def energy(self):
        return np.trapz(self.samples ** 2.0, self.t)

    def avg_power(self):
        return self.energy() / (np.max(self.t) - np.min(self.t))

    def fwm_at(self, L = -3):
        return level(self.t, self.samples, lvl = L, return_t = False)
#---signal2

class carrier(signal):
    """
    Carrier signal
    """
    def __init__(self, t, f0, A = 1, phi = 0.0):
        samples = A * np.cos( 2 * np.pi * f0 * t + phi)
        super().__init__( t = t, samples = samples )

class displaced_spulse(signal):
```

```
97
98      def __init__(self, t, T1, tcenter = 0.0):
99          samples = np.zeros(t.size)
100         i = np.where( ( (t - tcenter) >= 0 ) & ( (t-tcenter) < T1) )
101         samples[ i ] = 1
102         super().__init__( t = t, samples = samples )
103
104 class square_pulse(signal):
105     """
106     Square pulse
107     """
108     def __init__(self, t, T1, tcenter = 0.0):
109         samples = np.zeros(t.size)
110         i = np.where( np.abs(t - tcenter) <= 0.5 * T1)
111         samples[ i ] = 1
112         super().__init__( t = t, samples = samples )
113
114 #---impulse1
115 class impulse_signal(signal):
116     """
117     Impuse signal (delta)
118     """
119     def __init__(self, t, t0 = 0):
120         samples = np.zeros(t.size)
121         Dt = t[1] - t[0]
122         i = np.where( np.abs (t - t0) <= 0.5 * Dt)
123         samples[ i ] = 1 / Dt
124         super().__init__( t = t, samples = samples )
125 #---impulse2
126
127 #---digitalsignal1
128 class digital_signal(signal):
129
130     def __init__(self, TS = 1e-6, samples_per_symbol = 10,
131                  tinitial = 0, tguard = 0.0):
132
133         super().__init__()
134         self.TS = TS
135         self.samples_per_symbol = samples_per_symbol
136         self.tinitial = tinitial
137         self.tguard = tguard
138
139     def modulate_from_symbols( self, symbols ):
140
141         self.Tmin = self.tinitial - self.tguard
142         self.Tmax = self.tinitial + symbols.size * self.TS +  self.tguard
143         self.Dt = self.TS / self.samples_per_symbol
144         self.t = np.arange(self.Tmin, self.Tmax, self.Dt)
145         self.samples = np.zeros( self.t.size )
146         self.symbols = symbols
147         self.N = self.t.size
148
149         i = np.floor( (self.t - self.tinitial) / self.TS).astype(int)
150         j = np.where( np.logical_and(i >= 0, i < symbols.size ) )
151
152         self.samples[j] = symbols[ i[j] ]
```

```
In [3]: from commlib import signal

In [4]: s = signal()

In [5]: vars(s)
Out[5]: {'t': None, 'N': 0, 'Dt': None, 'samples': None}

In [6]:
```

Εικόνα 3.3: Δημιουργία ενός σήματος χωρίς αρχικές παραμέτρους.

```
153 #---digitalsignal2
```

Listing 3.1: Η πρώτη έκδοση του commlib.py

Η κλάση `signal` περιέχει δύο βασικά χαρακτηριστικά: τον άξονα του χρόνου t που στην ουσία περιέχει τις διακριτές χρονικές στιγμές t_i που αποθηκεύουμε το σήμα μας και τα δείγματα του σήματος στις χρονικές αυτές στιγμές, $x_i = x(t_i)$. Τα t_i τα αποθηκεύουμε στο χαρακτηριστικό `t` της κλάσης ενώ τα x_i στο χαρακτηριστικό `samples`.

Ο ορισμός της κλάσης περιλαμβάνει την μέθοδο `__init__` η οποία καλείται κάθε φορά που δημιουργείται η κλάση. Η μέθοδος αυτή δέχεται ως παραμέτρους την μεταβλητή `t` και την μεταβλητή `samples` οι οποίες έχουν προκαθορισμένη τιμή ίση με `None`. Το `None` το χρησιμοποιούμε όταν θέλουμε να δηλώσουμε ότι μία μεταβλητή δεν έχει τιμή, είναι δηλαδή κενή[1]. Κατά την δημιουργία της κλάσης θέτουμε το χαρακτηριστικό της `t` να είναι ίσο με την μεταβλητή `t`. Αυτό γράφεται ως:

```
1 self.t = t
```

Στη συνέχεια ελέγχουμε να δούμε εάν ο άξονας του χρόνου `self.t` είναι `None`. Στην περίπτωση που είναι τότε αυτό σημαίνει ότι ο χρήστης δεν μας προσδιόρισε τον άξονα του χρόνου. Αν όμως τον έχει προσδιορίσει τότε υπολογίζουμε την διαφορά μεταξύ δύο διαδοχικών χρονικών στιγμών και την αποθηκεύουμε στο χαρακτηριστικό `Dt` της κλάσης

```
1 self.Dt = t[1] - t[0]
```

καθώς επίσης και το πλήθος των σημείων των του άξονα του χρόνου t_i που το αποθηκεύουμε στο χαρακτηριστικό `N` της κλάσης.

```
1 self.N = t.size
```

Ας δούμε τώρα μερικά παραδείγματα του πως χρησιμοποιούμε την κλάση `signal`.

Στην εικόνα 3.3 δείχνουμε τι συμβαίνει όταν δημιουργούμε ένα σήμα χωρίς αρχικές παραμέτρους. Γράφοντας

```
1 s = signal()
```

οι παράμετροι `t` και `samples` δεν ορίζονται οπότε λαμβάνουν τις προκαθορισμένες τιμές που σύμφωνα με τον ορισμό της `__init__` είναι `None`. Όταν γράφουμε

```
1 vars(s)
```

[1]Είναι ενδιαφέρον να σημειώσουμε ότι ο Tony Hoare που δημιούργησε μεταξύ άλλων την γλώσσα ALGOL60 αποκαλεί το NULL από το οποίο είναι εμπνευσμένο το `None` το μεγαλύτερο του σφάλμα. Ας προσέξουμε επομένως πως το χρησιμοποιούμε!

```
In [12]: from commlib import signal

In [13]: t = np.linspace(0, 1, 10)

In [14]: t
Out[14]:
array([0.        , 0.11111111, 0.22222222, 0.33333333, 0.44444444,
       0.55555556, 0.66666667, 0.77777778, 0.88888889, 1.        ])

In [15]: t.size
Out[15]: 10

In [16]: samples = np.random.rand(t.size)

In [17]: samples
Out[17]:
array([0.41718111, 0.4890912 , 0.97875967, 0.85691597, 0.26325279,
       0.89621874, 0.17116009, 0.00779644, 0.05681521, 0.13967879])

In [18]: s = signal(t = t, samples = samples)

In [19]: vars(s)
Out[19]:
{'t': array([0.        , 0.11111111, 0.22222222, 0.33333333, 0.44444444,
        0.55555556, 0.66666667, 0.77777778, 0.88888889, 1.        ]),
 'Dt': 0.1111111111111111,
 'N': 10,
 'samples': array([0.41718111, 0.4890912 , 0.97875967, 0.85691597, 0.26325279,
        0.89621874, 0.17116009, 0.00779644, 0.05681521, 0.13967879])}
```

Εικόνα 3.4: Δημιουργία ενός σήματος με αρχικές παραμέτρους.

τότε μπορούμε να δούμε τις τιμές των χαρακτηριστικών που υπάρχουν μέσα στην κλάση. Όπως θα περιμέναμε, το `t`, το `Dt` και το `samples` είναι ίσα με `None` αφού δεν τα έχουμε ορίσει και το N είναι ίσο με μηδέν. Στην εικόνα 3.4 δείχνουμε τι συμβαίνει όταν αρχικοποιούμε την κλάση με κάποιες αρχικές παραμέτρους `t` και `samples`. Δημιουργούμε ένα πίνακα `numpy`, το `t` το οποίο ορίζεται ως εξής:

```
t = np.linspace(0, 1, 10)
```

Η `numpy.linspace` δημιουργεί ένα πίνακα `numpy` που ξεκινάει από το πρώτο όρισμα (δηλαδή το 0 στην συγκεκριμένη περίπτωση) και φτάνει ως το δεύτερο όρισμα (δηλαδή το 1) και έχει αριθμό στοιχείων ίσο με το τρίτο όρισμα (δηλαδή το 10). Τα στοιχεία του πίνακα t_i σχηματίζονται έτσι ώστε να ισαπέχουν, οπότε το $t_{i+1} - t_i$ θα είναι το ίδιο για όλα τα στοιχεία. Αν θέσουμε

$$t_{i+1} - t_i = \Delta t \tag{3.4}$$

τότε προκύπτει ότι

$$t_i = t_0 + i\Delta t \tag{3.5}$$

Αν ο πίνακας έχει N στοιχεία τότε το i πηγαίνει από το 0 εώς το $N-1$ οπότε μπορούμε επομένως να γράψουμε

$$t_{N-1} = t_0 + (-1)\Delta t \tag{3.6}$$

από όπου προκύπτει ότι:

$$\Delta t = \frac{t_{N-1} - t_0}{N-1} \tag{3.7}$$

Στην περίπτωση μας $N = 10$, $t_0 = 0$ και $t_{N-1} = 1$ οπότε $\Delta t = 0.111\cdots$ όπως φαίνεται και στην εικόνα 3.3.

Σε ότι αφορά την παράμετρο `samples`, χρησιμοποιούμε την `numpy.random.rand` για να θέσουμε τις αρχικές της τιμές. Η συνάρτηση αυτή θέτει τιμές που επιλέγονται τυχαία στο διάστημα $[0, 1]$.

Αυτό που έχουμε κάνει μέχρι τώρα είναι να φτιάξουμε ένα σήμα με $N = 10$ δείγματα $x_i = x(t_i)$ που αντιστοιχούν σε χρονικές τιμές $t_i = t_0 + i\Delta t$ όπου $0 \leq i \leq N-1$ με $t_0 = 0$ και $t_{N-1} = 1$.

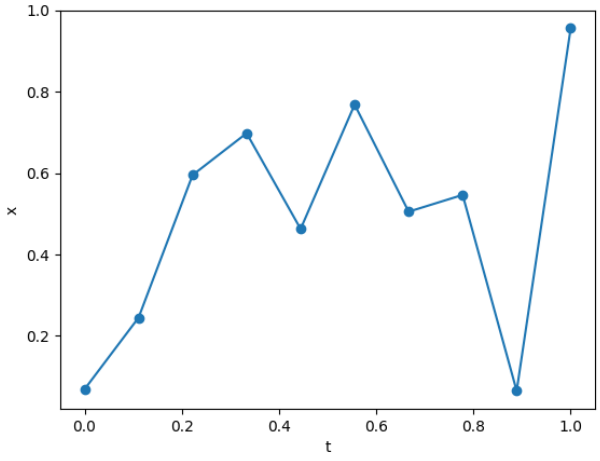
3.4 Γραφική παράσταση σήματος

Συχνά είναι χρήσιμο να κάνουμε γραφική παράσταση των σημάτων που φτιάχνουμε. Επομένως θα προσθέσουμε μία μέθοδο στην κλάση μας που θα την πούμε plot και η οποία θα αναλάβει ακριβώς αυτό. Το `commlib.py` τώρα περιέχει τα εξής:

```
import numpy as np
from matplotlib.pyplot import plt

class signal:
    """
    The basic signal class
    """
    def __init__(self, t = None, samples = None):
        """
        Initialize the signal class
        """
        self.t = t
        if self.t is not None:
            self.Dt = t[1] - t[0]
            self.N = t.size
        else:
            self.N = 0
            self.Dt = None

        self.samples = samples

    def plot(self, line_type = '-o'):
        """
        Plot the signal
        """
        plt.plot(self.t, self.samples, line_type)
        plt.xlabel('t')
        plt.ylabel('x')
```

Listing 3.2: Η μέθοδος `plot` της κλάσης `signal`

Η μέθοδος `plot` βασίζεται στη βιβλιοθήκη `matplotlib` και συγκεκριμένα στην κλάση `matplotlib.pyplot.plt` η οποία χρησιμοποιείται για γραφικές παραστάσεις τέτοιου είδους. Στην μέθοδο περνάμε ως παράμετρο και το `line_type` που καθορίζει το είδος της γραμμής που θα χρησιμοποιηθεί. Το `-o` σημαίνει ότι θα αποτελείται από κυκλάκια που θα ενώνονται με γραμμές. Στην πιο βασική της χρήση, η μέθοδος `matplotlib.pyplot.plt.plot` δέχεται ως πρώτη παράμετρο τις τιμές στον οριζόντιο άξονα (δηλαδή στην συγκεκριμένη περίπτωση τα t_i), ως δεύτερη παράμετρο τις τιμές στον κάθετο



Εικόνα 3.5: Γραφική παράσταση του σήματος.

άξονα (στην περίπτωση μας τα x_i) και ως τρίτη παράμετρο το είδος της γραμμής που όπως είδαμε καθορίζεται από το `line_type`.

Παρακάτω δείχνουμε πως μπορούμε να δοκιμάσουμε την κλάση `signal` κάνοντας γραφική παράσταση του σήματος που παράγεται.

```
import numpy as np
from commlib import signal
import matplotlib.pyplot as plt

t = np.linspace(0, 1, 10)
x = np.random.rand( t.size )
s = signal(t = t, samples = x)

plt.close('all')
plt.figure()
s.plot()
```

Listing 3.3: Γραφική παράσταση σημάτων.

3.5 Παραδείγματα σημάτων

Θα χρησιμοποιήσουμε τώρα την κλάση `signal` για να κατασκευάσουμε τετραγωνικούς παλμούς και φέροντα σήματα. Η `commlib.py` τώρα περιέχει δύο νέες κλάσεις όπως φαίνεται στο παρακάτω listing.

```
import numpy as np
from matplotlib.pyplot import plt

class signal:
    """
    The basic signal class
    """
    def __init__(self, t = None, samples = None):
```

```
        """
        Initialize the signal class
        """
        self.t = t
        if self.t is not None:
            self.Dt = t[1] - t[0]
            self.N = t.size
        else:
            self.N = 0
            self.Dt = None

        self.samples = samples

    def plot(self, line_type = '-o'):
        """
        Plot the signal
        """
        plt.plot(self.t, self.samples, line_type)
        plt.xlabel('t')
        plt.ylabel('x')

class carrier(signal):
    """
    Carrier signal
    """
    def __init__(self, t, f0, A = 1, phi = 0.0):
        samples = A * np.cos( 2 * np.pi * f0 * t + phi)
        super().__init__( t = t, samples = samples )

class square_pulse(signal):
    """
    Square pulse
    """
    def __init__(self, t, T1, tcenter = 0.0):
        samples = np.zeros(t.size)
        i = np.where( np.abs (t - tcenter) <= 0.5 * T1)
        samples[ i ] = 1
        super().__init__( t = t, samples = samples )
```

Listing 3.4: Οι κλάσεις `carrier` και `square_pulse`.

Η λογική που θα ακολουθήσουμε είναι να κληρονομήσουμε στις κλάσεις χαρακτηριστικά και μεθόδους από την γενική κλάση `signal`. Για παράδειγμα γράφοντας ότι:

```
class carrier(signal):
```

δηλώνουμε ότι η κλάση `carrier` που θα περιγράφει το φέρον σήμα έχει ως *γονέα* την κλάση `signal` και επομένως κληρονομεί όλες τις μεθόδους και όλα τα χαρακτηριστικά της. Στη συνέχεια βλέπουμε ότι η κλάση `carrier` ορίζει την δική της `__init__` επειδή τώρα μας ενδιαφέρει να δημιουργήσουμε τα δείγματα σύμφωνα με την (3.2) όπως δείχνει και το τμήμα του κώδικα

```
samples = A * np.cos( 2 * np.pi * f0 * t + phi )
```

Στο παραπάνω κώδικα χρησιμοποιούμε την `numpy.cos` για να υπολογίσουμε το συνημίτονο. Το `numpy.pi` είναι το γνωστό μας $\pi = 3.14\cdots$. Προσέξτε ότι οι παράμετροι της `__init__` είναι διαφορετικές, καθώς πέρα από τον άξονα του χρόνου `t` θα πρέπει να δώσουμε την

συχνότητα f_0 του φέροντος στην παράμετρο `f0`. Επίσης μπορούμε να καθορίσουμε το πλάτος A και την αρχική φάση ϕ μέσω των παραμέτρων `A` και `phi`. Αν δεν καθοριστούν τιμές για το A και ϕ θα θεωρήσουμε ότι $A = 1$ και $\phi = 0$. Ένα επίσης ενδιαφέρον σημείο είναι ότι η παρακάτω γραμμή κώδικα που υπάρχει στην `__init__`:

```
super().__init__( t = t, samples = samples )
```

Εδώ καλούμε το `__init__` του γονέα (δηλαδή της `signal`) για να αρχικοποιηθεί από εκεί και ύστερα η κλάση με τον ίδιο ακριβώς τρόπο όπως ο γονέας, δηλαδή να αποθηκευθεί ο άξονας του χρόνου και τα δείγματα του σήματος που τώρα όμως έχουν δημιουργηθεί σύμφωνα με την (3.2).

Η λογική της κλάσης `square_pulse` είναι παρόμοια μόνο που τώρα χρησιμοποιείται η (3.1) αντί της (3.2). Αλλά η υλοποίηση είναι λίγο διαφορετική. Αρχικά δημιουργούμε δείγματα του σήματος τα οποία είναι παντού μηδέν με την `numpy.zeros`:

```
samples = np.zeros(t.size)
```

Στη συνέχεια ψάχνουμε να βρούμε ποια δείγματα είναι μέσα στην διάρκεια του παλμού και επομένως πρέπει να γίνουν ίσα. Γράφουμε λίγο διαφορετικά την συνθήκη του πάνω κλάδου της (3.1),

$$|t| \leq \frac{T_1}{2} \tag{3.8}$$

όπου το $|t|$ είναι η απόλυτη τιμή του t που υπολογίζεται με την συνάρτηση `numpy.abs`. Η γραμμή:

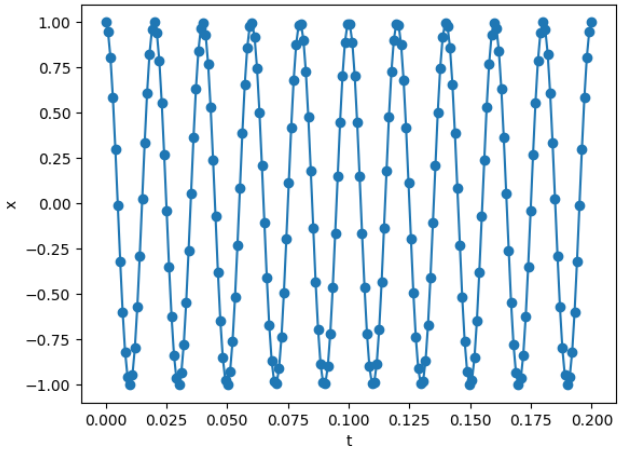
```
i = np.where( np.abs (t - tcenter) <= 0.5 * T1)
```

βρίσκει τις θέσεις που ισχύει η (3.8). Αν παρατηρήσετε καλά την παραπάνω γραμμή κώδικα υπάρχει μία μικρή προσθήκη στην συνθήκη, δηλαδή το `tcenter`. Η παράμετρος αυτή μας επιτρέπει να κεντράρουμε τον παλμό στην χρονική στιγμή που θέλουμε. Στη συνέχεια βάζουμε στις θέσεις αυτές την τιμή των δειγμάτων ίσα με 1 επειδή βρίσκονται μέσα στην διάρκεια του παλμού με την γραμμή:

```
samples[ i ] = 1
```

Με το παρακάτω listing φτιάχνουμε και κάνουμε την γραφική παράσταση ενός φέροντος και ενός τετραγωνικού παλμού.

```
import matplotlib.pyplot as plt
import numpy as np
from commlib import carrier, square_pulse

f0c = 50
Tminc = 0
Tmaxc = 0.2
N = 200
t = np.linspace(Tminc, Tmaxc, N)

c = carrier(t, f0c)
plt.close('all')
plt.figure()
c.plot()

T1 = 0.05
```



Εικόνα 3.6: Παράδειγμα φέροντος.

```
tcenter = 0.1
p = square_pulse(t, T1, tcenter = tcenter)
plt.figure()
p.plot()
```

Listing 3.5: carrierpulseplot.py

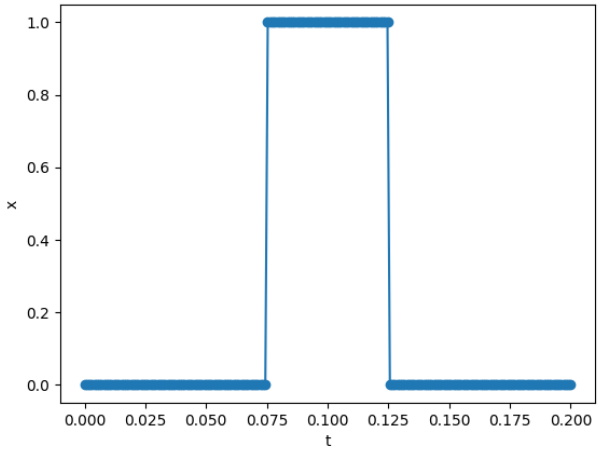
Στην εικόνα 3.6 δείχνουμε το φέρον σήμα που παράγεται από τον κώδικα ενώ στο 3.7 τον τετραγωνικό παλμό.

DRY - do not repeat your self

Όπως βλέπετε προσπαθούμε να εκμεταλλευτούμε στο έπακρο τις δυνατότητες που μας δίνει η κληρονομικότητα των κλάσεων. Σκοπός εδώ είναι να αποφύγουμε όσο το δυνατόν να γράφουμε τον ίδιο κώδικα ξανά και ξανά. Έτσι κάποια πράγματα που είναι κοινά σε όλα τα σήματα τα τοποθετούμε σε μία γενική κλάση `signal` και τα πιο συγκεκριμένα σήματα όπως το φέρον ή ο τετραγωνικός παλμός κληρονομούν αυτά τα χαρακτηριστικά και τις μεθόδους. Γράφουμε επομένως τον κοινό κώδικα μία φορά στην γενική κλάση. Θα ακολουθήσουμε αυτή την τακτική που ονομάζεται συχνά DRY (do not repeat yourself - μην επαναλαμβάνεσαι) όσο το δυνατόν περισσότερο.

3.6 Ψηφιακά σήματα

Όπως είδαμε στην ενότητα 3.2 και στην εικόνα 3.2, τα ψηφιακά σήματα αλλάζουν τιμές σε δεδομένες χρονικές στιγμές $t = t_i$. Οι τιμές του σήματος σε αυτές τις χρονικές στιγμές



Εικόνα 3.7: Παράδειγμα τετραγωνικού παλμού.

συμβολίζονται με x_i και συνήθως επιλέγονται από έναν πεπερασμένο σύνολο δυνατών τιμών $\mathscr{V} = \{v_1, v_2, \cdots, v_M\}$ όπου M είναι ο αριθμός των διαφορετικών δυνατών τιμών. Οι διαφορετικές τιμές v_k ονομάζονται συνήθως *σύμβολα* του σήματος και το $\mathscr{V}$ ονομάζεται *αστερισμός συμβόλων*. Επομένως κάθε ένα από τα x_i μπορεί να ισούται με ένα σύμβολο v_k από το $\mathscr{V}$ και μόνο από αυτό.

Τα ψηφιακά σήματα παράγονται στην έξοδο κατάλληλων ηλεκτρονικών κυκλώματα. Ωστόσο στη φύση τα σήματα είναι αναλογικά οπότε τα εν λόγω κυκλώματα παράγουν μία αναλογική προσέγγιση $x(t)$ του ψηφιακού σήματος x_i. Οπότε καταρχήν για $t = t_i$, θα πρέπει να έχουμε $x(t_i) = x_i$. Τις υπόλοιπες όμως χρονικές στιγμές, $t \neq t_i$, πως είναι το σήμα $x(t)$ που παράγεται;

Ένα ιδανικό ψηφιακό κύκλωμα θα προσπαθούσε να αναπαράγει την συμπεριφορά του σήματος στην εικόνα 3.2. Αν υποθέσουμε ότι οι στιγμές t_i που το σήμα μπορεί να αλλάξει τιμή είναι ισαπέχουσες τότε $t_i = iT_S$ όπου το T_S ονομάζεται *διάρκεια συμβόλου*. Τότε θα θέλαμε το $x(t)$ να είναι σταθερό και ίσο με το εκάστοτε σύμβολο $x_i = v_k$ μέσα στο διάστημα $t \in [iT_S, (i+1)T_S)$, δηλαδή:

$$x(t) = x_i \quad , iT_S \leq t < (i+1)T_S \tag{3.9}$$

```
131 class digital_signal(signal):
132
133     def __init__(self, TS = 1e-6, samples_per_symbol = 10,
134                  tinitial = 0, tguard = 0.0):
135
136         super().__init__()
137         self.TS = TS
138         self.samples_per_symbol = samples_per_symbol
139         self.tinitial = tinitial
140         self.tguard = tguard
141
142     def modulate_from_symbols( self, symbols ):
```

```
143
144         self.Tmin = self.tinitial - self.tguard
145         self.Tmax = self.tinitial + symbols.size * self.TS +  self.tguard
146         self.Dt = self.TS / self.samples_per_symbol
147         self.t = np.arange(self.Tmin, self.Tmax, self.Dt)
148         self.samples = np.zeros( self.t.size )
149         self.symbols = symbols
150         self.N = self.t.size
151
152         i = np.floor( (self.t - self.tinitial) / self.TS).astype(int)
153         j = np.where( np.logical_and(i >= 0, i < symbols.size ) )
154
155         self.samples[j] = symbols[ i[j] ]
```

Listing 3.6: Η κλάση `digital_signal`

Στο listing 3.6 δείχνουμε την κλάση `digital_signal` η οποία χρησιμοποιείται για να δημιουργήσει την αναλογική προσέγγιση του ψηφιακού σήματος βάσει της (3.9). Η κλάση αρχικοποιείται ορίζοντας το T_S και τον αριθμό των δειγμάτων του αναλογικού σήματος που αντιστοιχούν σε ένα σύμβολο χρησιμοποιώντας την παράμετρο `samples_per_symbol`. Για παράδειγμα αν `samples_per_symbol = 10`, θα αντιστοιχούν 10 χρονικές στιγμές στον άξονα του χρόνου `self.t` για κάθε διάρκεια συμβόλου. Η παράμετρος `tinitial` καθορίζει την χρονική στιγμή t_0 στην οποία αρχίζει η διάρκεια του πρώτου συμβόλου. Τέλος η παράμετρος `tguard` χρησιμοποιείται για να καθορίσει ένα αρχικό διάστημα προστασίας, διάρκειας t_g, στο οποίο το σήμα θα είναι ίσο με μηδέν, πριν αρχίσουν τα σύμβολα. Έτσι θα θέσουμε $x(t) = 0$ για $t_0 - t_g \leq t < t_0$. Αν ο συνολικός αριθμός συμβόλων είναι N, τότε εισάγουμε και ένα ίσης διάρκειας διάστημα μετά την διάρκεια του τελευταίου συμβόλου. Δεδομένου ότι το τελευταίο σύμβολο διαρκεί από $t \leq t_0 + (N-1)T_S$ μέχρι $t < t_0 + NT_S$, το διάστημα προστασίας είναι $t_0 + NT_S \leq t < t_0 + NT_S + t_g$. Τα διαστήματα προστασίας χρησιμοποιούνται για να έχουμε καλύτερη αριθμητική προσέγγιση όταν υπολογίζουμε το φάσμα του σήματος (θα δούμε μερικά παραδείγματα σε άλλα κεφάλαια). Το σήμα που θα φτιάξουμε θα καθορίζεται από την παρακάτω σχέση:

$$x(t) = \begin{cases} 0 & ,t_0 - t_g \leq t < t_0 \\ x_i & ,t_0 + iT_S \leq t < t_0 + (i+1)T_S \\ 0 & ,t_0 + (N-1)T_S \leq t < t_0 + NT_S + t_g \end{cases} \tag{3.10}$$

Η μέθοδος `modulate_from_symbols` χρησιμοποιείται για να φτιάξει το ψηφιακό σήμα από τα σύμβολα που δίνονται στην παράμετρο `symbols`. Οι τιμές x_i στην (3.10) επομένως καθορίζονται από τα τιμές του πίνακα `symbols[i]`. Η `modulate_from_symbols` φτιάχνει τον άξονα του χρόνου υπολογίζοντας την μικρότερη χρονική στιγμή $t_0 - t_g$ στην (3.10),

```
1  self.Tmin = self.tinitial - self.tguard
```

Η μεγαλύτερη χρονική στιγμή $t_0 + NT_S - t_g$ υπολογίζεται στην συνέχεια,

```
1  self.Tmax = self.tinitial + symbols.size * self.TS - self.tguard
```

Στη συνέχεια θα πρέπει να υπολογίσουμε τα δείγματα του αναλογικού σήματος σύμφωνα με την (3.10). Η διάρκεια του i συμβόλου είναι $t_0 + iT_S \leq t < t_0 + (i+1)T_S$. Από την τελευταία σχέση προκύπτει:

$$i \leq \frac{t - t_0}{T_S} < i + 1 \tag{3.11}$$

Για κάθε πραγματικό αριθμό r, υπάρχει ένας ακέραιος n για τον οποίο $n \leq r < n+1$. Το n ονομάζεται *ακέραιο μέρος* του r και συμβολίζεται με $\lfloor x \rfloor$. Από την (3.11), προκύπτει ότι:

$$i = \left\lfloor \frac{t - t_0}{T_S} \right\rfloor \tag{3.12}$$

Στην `modulate_from_symbols` χρησιμοποιούμε την (3.12) ως εξής:

```
  i = np.floor( (self.t - self.tinitial) / self.TS).astype(int)
  j = np.where( np.logical_and(i >= 0, i < symbols.size ) )
```

Η πρώτη εντολή υπολογίζει έναν πίνακα `i` σύμφωνα με την (3.12), που περιέχει για κάθε χρονική στιγμή που βρίσκεται στον άξονα του χρόνου του σήματος (δηλαδή το `self.t`) τον αύξων αριθμό του συμβόλου στο οποίο αντιστοιχεί η χρονική στιγμή αυτή. Η δεύτερη εντολή κρατάει τις τιμές του `i` που είναι ανάμεσα στο 0 και στο $N-1$ όπου N είναι το πλήθος των συμβόλων που παρέχονται ως είσοδος με την μεταβλητή `symbols`.

```
from commlib import digital_signal
import numpy as np

TS = 1e-6
symbols = np.array([-1, 1, -1, 1, -1, 1])
tguard = 3 * TS
samples_per_symbol = 5
tinitial = 0

x = digital_signal(TS = TS,
                   samples_per_symbol = samples_per_symbol,
                   tinitial = tinitial,
                   tguard = tguard)

x.modulate_from_symbols( symbols )
x.plot()
```

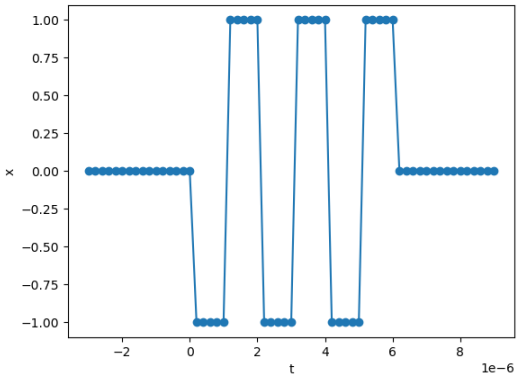
Listing 3.7: `digtest.py`

Στο listing 3.7, δείχνουμε ένα παράδειγμα χρήσης της `digital_signal`. Θεωρούμε περίοδο συμβόλου $T_S = 1$ μs, πέντε δείγματα ανά διάρκεια συμβόλου (`samples_per_symbol = 5`) και πως έχουμε 6 σύμβολα τα οποία προκύπτουν από εναλλαγή των τιμών -1 και $+1$. Θέτουμε διάστημα προστασίας ίσο με 3 διάρκειες συμβόλων (`tguard = 3 * TS`) και πως το πρώτο σύμβολο ξεκινά την χρονική στιγμή $t = 0$ (`tinitial = 0`). Στην εικόνα 3.8 δείχνουμε το ψηφιακό σήμα που παράγεται από το listing 3.7.

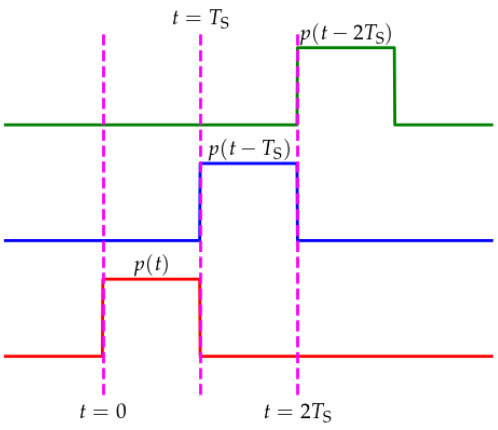
Η (3.9) και η επέκταση της (3.10) δείχνουν τον τρόπο για την υλοποίηση της συνάρτησης `digital_signal`. Ωστόσο για τις μαθηματικές αναλύσεις είναι πιο χρήσιμο να μην έχουμε σήματα με πολλαπλούς κλάδους όπως στις (3.9) και (3.10). Ας θεωρήσουμε έναν τετραγωνικό παλμό $p(t)$ που είναι ίσος με 1 από το $t = 0$ μέχρι $t = T_S$,

$$p(t) = \begin{cases} 1 & ,0 \leq t < T_S \\ 0 & ,\text{διαφορετικά} \end{cases} \tag{3.13}$$

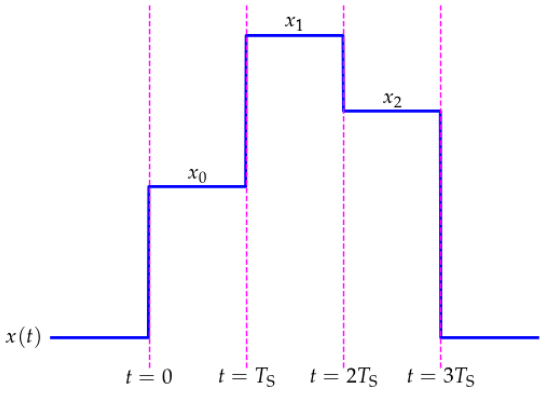
Για το $p(t)$ δίνεται από την (3.13) ο παλμός ξεκινάει από το $t = 0$ και διαρκεί ως το $t = T_S$. Το σήμα $p(t - kT_S)$ είναι ο ίδιος τετραγωνικός παλμός που ξεκινάει όμως από το $t = kT_S$ και διαρκεί ως το $t = (k+1)T_S$, είναι δηλαδή μετατοπισμένος κατά kT_S στον άξονα του χρόνου. Στην εικόνα 3.9, δείχνουμε τους μετατοπισμένους παλμούς $p(t - kT_S)$ για $k = 0, 1$ και 2.



Εικόνα 3.8: Παράδειγμα χρήσης της `digital_signal`



Εικόνα 3.9: Μετατοπισμένοι τετραγωνικοί παλμοί $p(t - kT_S)$.



Εικόνα 3.10: Το σήμα $x(t)$ στην (3.14).

Τόσο από την εικόνα 3.9 όσο και από την (3.13) παρατηρούμε ότι οι παλμοί δεν επικαλύπτονται. Ας θεωρήσουμε το σήμα:

$$x(t) = x_0 p(t) + x_1 p(t - T_S) + x_2 p(t - 2T_S) \tag{3.14}$$

το οποίο φαίνεται και στην εικόνα 3.10. Παρατηρούμε ότι εφόσον οι παλμοί δεν επικαλύπτονται θα έχουμε $x(t) = x_0$ για $0 \leq t < T_S$, $x(t) = x_1$ για $T_S \leq t < 2T_S$ και $x(t) = x_2$ για $2T_S \leq t < 3T_S$. Επομένως το πλάτος x_k μπροστά από κάθε παλμό $p(t - kT_S)$, καθορίζει την τιμή του σήματος $x(t) = x_k$, στο διάστημα $kT_S \leq t < (k+1)T_S$. Αν θεωρήσουμε περισσότερους όρους στο άθροισμα (3.14), δηλαδή:

$$x(t) = \sum_{k=0}^{N} x_k p(t - kT_S) \tag{3.15}$$

τότε με την ίδια λογική μπορούμε να δείξουμε ότι:

$$x(t) = x_k \quad \text{όταν } kT_S \leq t < (k+1)T_S \tag{3.16}$$

Η (3.16) στην ουσία ταυτίζεται με την (3.9) και επομένως η (3.15) αποτελεί μία αναπαράσταση του ψηφιακού σήματος χωρίς να χρησιμοποιούμε εξισώσεις με κλάδους. Θα την χρησιμοποιήσουμε στα παρακάτω κεφάλαια για τις θεωρητικές μας αναλύσεις.

3.7 Ενέργεια και ισχύς

Αν θεωρήσουμε ένα σήμα τάσης $v(t)$ το οποίο εφαρμόζεται στα άκρα μίας αντίστασης R τότε το ρεύμα που την διαρρέει είναι

$$i(t) = \frac{v(t)}{R} \tag{3.17}$$

Σύμφωνα με τα όσα γνωρίζουμε από την ηλεκτρονική, η *στιγμιαία* ισχύς $P_R(t)$ που καταναλώνεται πάνω στην R, δίνεται από την σχέση:

$$P_R(t) = v(t)i(t) = \frac{v^2(t)}{R} \tag{3.18}$$

Σύμφωνα με την (3.18), η στιγμιαία ισχύς είναι ανάλογη του τετραγώνου του σήματος τάσης. Επομένως έχει νόημα να ορίσουμε την στιγμιαία ισχύ $P(t)$ ενός οποιοδήποτε σήματος $x(t)$ ως εξής:

$$P(t) = x^2(t) \tag{3.19}$$

Η *ενέργεια* $E(t_a,t_b)$ ενός σήματος μέσα στην χρονική διάρκεια $[t_a,t_b]$ είναι απλά το ολοκλήρωμα της ισχύος,

$$E(t_a,t_b) = \int_{t_a}^{t_b} P(t)\mathrm{d}t \tag{3.20}$$

Παράδειγμα 3.7.1: **Ενέργεια φέροντος σήματος**

Δίνεται το σήμα:

$$x(t) = A\cos(2\pi f_0 t) \tag{3.21}$$

με $f_0 = 1\text{MHz}$ και $A = 1\sqrt{\text{W}}$. Το παραπάνω σήμα ονομάζεται *φέρον* σήμα. Να υπολογίσετε την ενέργεια του σήματος $E(0,T)$ μέσα στο διάστημα $[0,T]$ όπου $T = 2/f_0$.

Λύση

Απλά εφαρμόζουμε τον τύπο (3.20). Θα έχουμε [a]:

$$E(0,T) = \int_0^T A^2\cos^2(2\pi f_0 t)\mathrm{d}t = A^2\int_0^T \cos^2(2\pi f_0 t)\mathrm{d}t =$$
$$A^2\left[\frac{t}{2} + \frac{1}{4\pi f_0}\sin(4\pi f_0 t)\right]_0^T = \frac{A^2T}{2} + \frac{A^2}{4\pi f_0}\sin(4\pi f_0 T) = A^2\frac{T}{2} = A^2 f_0 \tag{3.22}$$

Αντικαθιστώντας βρίσκουμε ότι $E(0,T) = 10^{-6}\text{Ws} = 10^{-6}\text{J}$

[a] Το ολοκλήρωμα $\int\cos^2(at)\mathrm{d}t$ ισούται με $\frac{t}{2} + \frac{\sin(2at)}{4a}$

2

[2] Θυμηθείτε από την φυσική που μάθαμε στο σχολείο, η ισχύς P μετριέται σε Watt ενώ η ενέργεια $E(t_a,t_b)$ σε Joule. Αν η ισχύς P είναι σταθερή με το χρόνο, τότε $E(t_a,t_b) = P(t_b - t_a)$

Η *μέση* ισχύς $\bar{P}(t_a, t_b)$ υπολογίζεται ως εξής:

$$\bar{P}(t_a, t_b) = \frac{E(t_a, t_b)}{t_b - t_a} \tag{3.23}$$

Στην περίπτωση του σήματος στο παράδειγμα 3.7.1 η μέση σε μία περίοδο $[0, T_0]$ όπου $T_0 = 1/f_0$ προκύπτει ως εξής:

$$\bar{P}(0, T_0) = \frac{E(0, T_0)}{T_0} = \frac{A^2}{2} \tag{3.24}$$

Συχνά θεωρούμε ότι $t_a \to -\infty$ και $t_b \to +\infty$ οπότε μπορούμε υπολογίζουμε την μέση ισχύ στο διάστημα $(-\infty, +\infty)$,

$$\bar{P} = \lim_{\tau \to \infty} \frac{1}{\tau} \int_{-\tau/2}^{\tau/2} |x(t)|^2 \mathrm{d}t \tag{3.25}$$

Στην περίπτωση ενός περιοδικού σήματος $x(t)$ μπορούμε να δείξουμε ότι η μέση ισχύς $\bar{P}$ στο $(-\infty, +\infty)$ ισούται με την μέση ισχύ στην περίοδο του σήματος $[0, T]$ όπου T είναι η περίοδος του σήματος. Υπάρχει μία πολύ χρήσιμη ιδιότητα για να υπολογίζουμε όρια της μορφής $\lim_{\tau \to \infty} f(\tau)$ όπως η (3.25). Αν θεωρήσουμε μία ακολουθία τ_n για την οποία ισχύει

$$\lim_{n \to \infty} \tau_n = +\infty \tag{3.26}$$

Τότε θα έχουμε:

$$\lim_{\tau \to \infty} f(\tau) = \lim_{n \to \infty} f(\tau_n) \tag{3.27}$$

Στην περίπτωση της (3.25) μπορούμε να θέσουμε $\tau_n = 2nT$ οπότε όταν $n \to \infty$ θα έχουμε $\tau_n \to \infty$. Το ολοκλήρωμα γράφεται ως εξής:

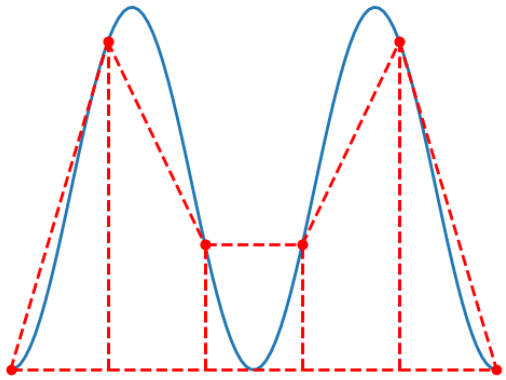
$$\frac{1}{\tau_n} \int_{-\tau_n/2}^{\tau_n/2} |x(t)|^2 \mathrm{d}t = \frac{1}{2nT} \int_{-nT}^{nT} |x(t)|^2 \mathrm{d}t \tag{3.28}$$

Το ολοκλήρωμα στο διάστημα $[-nT, nT]$ μπορεί να σπάσει σε $2n$ ολοκληρώματα σε διαστήματα με μήκος μία περίοδο, της μορφής $[mT, (m+1)T]$. Εφόσον το $x(t)$ είναι περιοδικό το ολοκλήρωμα του $|x(t)|^2$ θα είναι το ίδιο σε όλα αυτά τα διαστήματα, οπότε:

$$\int_{-nT}^{nT} |x(t)|^2 \mathrm{d}t = \underbrace{\int_{-nT}^{-(n-1)T} |x(t)|^2 \mathrm{d}t + \cdots + \int_{mT}^{(m+1)T} |x(t)|^2 \mathrm{d}t + \cdots + \int_{(n-1)T}^{nT} |x(t)|^2 \mathrm{d}t}_{2n \text{ όροι}}$$
$$= 2n \int_0^T |x(t)|^2 \mathrm{d}t \tag{3.29}$$

Για το όριο μπορούμε να γράψουμε:

$$\bar{P} = \lim_{\tau \to \infty} \frac{1}{\tau} \int_{-\tau/2}^{\tau/2} |x(t)|^2 \mathrm{d}t = \lim_{n \to \infty} \frac{1}{2nT} \int_{-nT}^{nT} |x(t)|^2 \mathrm{d}t = \frac{1}{T} \int_0^T |x(t)|^2 \mathrm{d}t = \bar{P}(0, T) \tag{3.30}$$



Εικόνα 3.11: Ο κανόνας του τραπεζίου.

Επομένως η μέση ισχύς $\bar{P}$ ενός περιοδικού σήματος $x(t)$ στο $(-\infty, +\infty)$ ισούται με την μέση ισχύ $\bar{P}(0,)$ μέσα στην βασική του περίοδο $[0, T]$. Στην περίπτωση του σήματος στο παράδειγμα 4.6.2 θα έχουμε:

$$\bar{P} = \bar{P}(0,) = \frac{A^2}{2} \tag{3.31}$$

Θα ήταν χρήσιμο να έχουμε έναν τρόπο να υπολογίζουμε την ισχύ και την ενέργεια και στην `Python`. Για να κάνουμε κάτι τέτοιο όμως, θα πρέπει να μπορούμε να υπολογίζουμε το ολοκλήρωμα της 3.20. Ευτυχώς η `numpy` μας δίνει έναν σχετικά απλό τρόπο να το κάνουμε χρησιμοποιώντας την `numpy.trapz` η οποία βασίζεται στον κανόνα του τραπεζίου.

```
52 class signal:
53     """
54     The basic signal class
55     """
56     def __init__(self, t = None, samples = None):
57         """
58         Initialize the signal class
59         """
60         self.t = t
61         if self.t is not None:
62             self.Dt = t[1] - t[0]
63             self.N = t.size
64         else:
65             self.N = 0
66             self.Dt = None
67 
68         self.samples = samples
69 
70     def plot(self, line_type = '-o', label = None):
71         """
72         Plot the signal
```

```
        """
        plt.plot(self.t, self.samples, label = label)
        plt.xlabel('t')
        plt.ylabel('x')

    def energy(self):
        return np.trapz(self.samples ** 2.0, self.t)

    def avg_power(self):
        return self.energy() / (np.max(self.t) - np.min(self.t))

#---fwmat1
    def fwm_at(self, L = -3):
        return level(self.t, self.samples, lvl = L, return_t = False)
#---fwmat2
```

Listing 3.8: Υπολογισμός ενέργειας και μέσης ισχύος

Στην εικόνα 3.11 δείχνουμε ένα παράδειγμα εφαρμογής του κανόνα του τραπεζίου όπου προσεγγίζουμε το εμβαδόν μίας συνάρτησης (μπλε καμπύλη) με το εμβαδόν των τραπεζίων τα οποία σχηματίζονται από διαδοχικά σημεία της συνάρτησης (κόκκινες διακεκομμένες γραμμές). Κάτι ανάλογο κάνει και η `numpy.trapz`. Στο listing 3.8 βλέπουμε την χρήση της `numpy.trapz` για τον υπολογισμό της ενέργειας στην μέθοδο `signal.energy` η οποία φυσικά κληρονομείται και στις κλάσεις `carrier` και `square_pulse`. Στην `numpy.trapz` περνάμε δύο ορίσματα. Το δεύτερο περιέχει τις τιμές του άξονα στον οποίο γίνεται η ολοκλήρωση και το πρώτο οι αντίστοιχες τιμές της συνάρτησης. Στην δική μας περίπτωση η ολοκλήρωση γίνεται ως προς το χρόνο t και αποθηκεύουμε συγκεκριμένες τιμές του άξονα $t = t_i$ στο χαρακτηριστικό `t` της κλάσης `signal` το οποίο μέσα στην κλάση αναφέρεται ως `self.t`. Το πρώτο όρισμα είναι οι τιμές της συνάρτησης, δηλαδή στην περίπτωση μας τα τετράγωνα $x^2 = x^2(t_i)$ τα οποία αποθηκεύονται στο χαρακτηριστικό `samples` της κλάσης που στο εσωτερικό της κλάσης αναφέρεται ως `self.samples`. Οπότε η γραμμή

```
return np.trapz(self.samples ** 2.0, self.t)
```

στην ουσία επιστρέφει προσεγγιστικά το ολοκλήρωμα της συνάρτησης $x^2(t)$ από το t_0 έως το t_{N-1}, δηλαδή:

$$\int_{t_0}^{t_{N-1}} x^2(t)\mathrm{d}t \tag{3.32}$$

Θα πρέπει να σημειώσουμε ότι η προσέγγιση του ολοκληρώματος είναι καλύτερη όσο περισσότερα εμβαδά θεωρήσουμε στην Εικόνα 3.11.

Παράδειγμα 3.7.2: Αριθμητικός υπολογισμός της ενέργειας του σήματος

Δίνεται το σήμα:

$$x(t) = A\cos(2\pi f_0 t) \tag{3.33}$$

με $f_0 = 1\text{MHz}$ και $A = 1\sqrt{\text{W}}$. Το παραπάνω σήμα ονομάζεται *φέρον* σήμα. Να υπολογίσετε την ενέργεια του σήματος $E(0,T)$ μέσα στο διάστημα $[0,T]$ όπου $T = 2/f_0$,

```
Energy: 1.000000e-06
Power: 5.000000e-01
```

Εικόνα 3.12: Υπολογισμός της μέσης ισχύος και ενέργειας.

χρησιμοποιώντας την Python

Λύση

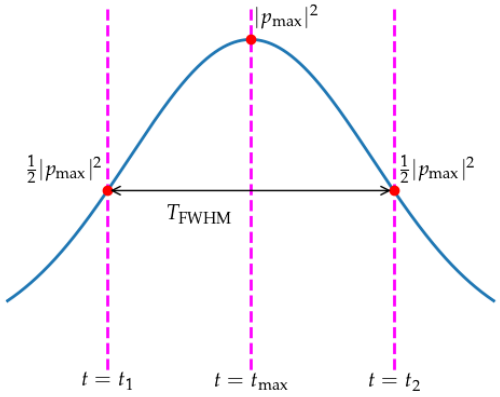
Χρησιμοποιούμε το listing 3.9 για να υπολογίσουμε αριθμητικά την ενέργεια και την μέση ισχύ του σήματος. Στην εικόνα 3.12 δείχνουμε το αποτέλεσμα των υπολογισμών. Σε ότι αφορά την ενέργεια λαμβάνουμε 10^{-6} που είναι σε πολύ καλή συμφωνία με τους θεωρητικούς υπολογισμούς του παραδείγματος 3.7.1.

```python
import numpy as np
from commlib import carrier
import matplotlib.pyplot as plt

Tmin = 0
f0 = 1e6
Tmax = 2 / f0
N = 100
t = np.linspace(Tmin, Tmax, N)
x = carrier(t, f0)

plt.close('all')
plt.figure(1)
x.plot()

E = x.energy()
P = x.avg_power()
print('Energy: %e' %E)
print('Power: %e' %P)
```

Listing 3.9: calcEP.py

3.8 Διάρκεια Παλμού

Όταν έχουμε να κάνουμε με ψηφιακά σήματα της μορφής (3.15), έχει ιδιαίτερη σημασία η διάρκεια των παλμών $p(t)$. Ωστόσο, θα δούμε ότι ο παλμός $p(t)$ δεν είναι απαραίτητα τετραγωνικός και είναι χρήσιμο να μπορούμε να μετρούμε το εύρος του παλμού. Στην εικόνα 3.13, δείχνουμε μία γενικότερη μορφή παλμού $p(t)$ και πως μπορούμε να μετρήσουμε την διάρκεια του χρησιμοποιώντας το *πλήρες εύρος μισής ισχύος* (full width half maximum - FWHM) το οποίο το συμβολίζουμε με T_{FWHM}. Στην ουσία θεωρούμε την *στιγμιαία* ισχύ του παλμού, $|p(t)|^2$, βρίσκουμε την χρονική στιγμή $t_{\max}$ που λαμβάνει τη μέγιστη τιμή της $|p_{\max}|^2$ και υπολογίζουμε τις χρονικές στιγμές t_1 και t_2 αριστερά και δεξιά το $t_{\max}$ όπου η ισχύς μειώνεται κατά 50% (−3dB), δηλαδή γίνεται $\frac{1}{2}|p_{\max}|^2$. Το T_{FWHM} δίνεται από την σχέση $T_{\text{FHWM}} = t_2 - t_1$ και είναι ένας τρόπος να



Εικόνα 3.13: Το πλήρες εύρος μισής ισχύος.

μετράμε το εύρος του παλμού, θα μπορούσαμε γενικότερα να υπολογίζουμε τις χρονικές στιγμές όπου η ισχύς του παλμού πέφτει κατά L dB από την $|p_{\max}|^2$ όπου το L δεν είναι απαραίτητα 3dB οπότε στην περίπτωση αυτή κάνουμε λόγο για το πλήρες εύρος του παλμού στα L dB. Στο listing 3.10 δείχνουμε την συνάρτηση `level` η οποία μπορεί να χρησιμοποιηθεί για τον υπολογισμό του πλήρους εύρους στα L dB. Βασισμένοι στην `level` υλοποιούμε και την μέθοδο `fwm_at` που υπολογίζει το πλήρες εύρος για το σήμα που περιγράφει η κλάση `signal`, όπως φαίνεται και στο listing 3.11. Στο listing 3.12, δείχνουμε τον υπολογισμό του T_{FWHM} ($L = 3$dB) για δύο περιπτώσεις παλμών $p(t)$.

Η λογική πίσω από την `level` είναι σχετικά απλή: βρίσκουμε τη θέση του μεγίστου του παλμού χρησιμοποιώντας την `numpy.argmax` η οποία μας επιστρέφει το δείκτη `imax` του πίνακα `x` στον οποίο βρίσκεται το μέγιστο του. Στη συνέχεια ψάχνουμε δεξιά και αριστερά του `imax` να βρούμε τους δείκτες `iright` και `ileft` όπου οι τιμές του `x` πέφτουν για πρώτη φορά `lvl` dB κάτω από το μέγιστο. Στη συνέχεια χρησιμοποιώντας την `numpy.interp` υπολογίζουμε με μεγαλύτερη ακρίβεια τη θέση όπου το $|p(t)|^2$ πέφτει κατά L dB κάτω από το μέγιστο.

```
5  def level(t, x, lvl = -3, post_only = False, return_t = False):
6      if post_only:
7          ipos = np.where(t >= 0)
8          x = x[ipos]
9          t = t[ipos]
10
11     X = np.abs(x) ** 2.0
12     XdB = 10 * np.log10(X)
13
14
15     imax = np.argmax( XdB )
16     Xmax = XdB[imax]
17     XdB_lvl = Xmax - lvl
18
19     i = imax
20     Xcur = XdB[i]
```

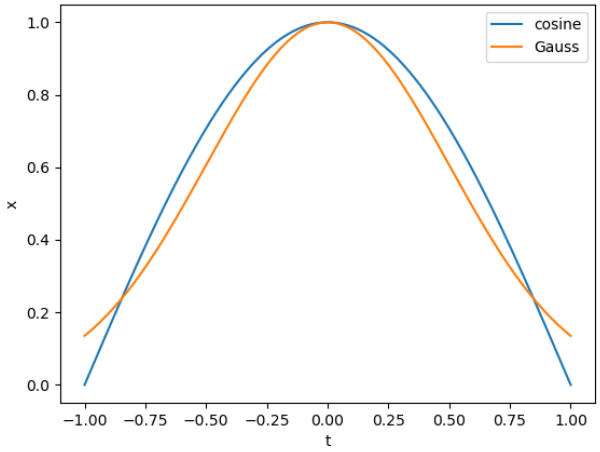
```
21
22      while (i < x.size) and ( Xcur >= XdB_lvl ):
23          i += 1
24          Xcur = XdB[i]
25      if (Xcur < XdB_lvl):
26          tright = np.interp(XdB_lvl, XdB[imax:(i+1)], t[imax:(i+1)] )
27      else:
28          tright = np.Inf
29
30
31      i = imax
32      Xcur = XdB[i]
33
34      while (i >= 0) and ( Xcur >= XdB_lvl ):
35          i -= 1
36          Xcur = XdB[i]
37
38      if (Xcur < XdB_lvl):
39          tleft = np.interp(XdB_lvl, XdB[i:imax], t[i:imax] )
40      else:
41          tleft = -np.Inf
42
43      if return_t:
44          return {'tleft' : tleft,
45                  'tright' : tright,
46                  'B' : tright - tleft}
47      else:
48          return tright - tleft
```

Listing 3.10: η συνάρτηση `level`

```
85      def fwm_at(self, L = -3):
86          return level(self.t, self.samples, lvl = L, return_t = False)
```

Listing 3.11: η συνάρτηση `fwm_at`

```
1  from commlib import signal
2  import numpy as np
3  import matplotlib.pyplot as plt
4
5  T1 = 1
6  Nt = 1000
7  L = 3
8  t = np.linspace(-T1, T1, Nt)
9  samples = np.cos( np.pi / T1 / 2 * t )
10 s = signal(t = t, samples = samples)
11
12 plt.close('all')
13 s.plot(label = 'cosine')
14 B3dB = s.fwm_at(L = L)
15 print('FWHM for cosine pulse: ', B3dB)
16
17 Tg = 0.5
18 samples = np.exp( -t ** 2.0 / 2 / Tg ** 2.0 )
19 s = signal(t = t, samples = samples)
20 s.plot(label = 'Gauss')
21 B3dB = s.fwm_at(L = L)
```



Εικόνα 3.14: Τα παραδείγματα παλμών στο listing 3.12.

```
print('FWHM for Gaussian pulse: ', B3dB)

plt.legend()
```

Listing 3.12: `levelex.py`

Το listing 3.12 κάνουμε και γραφική παράσταση των παλμών που χρησιμοποιούμε η οποία φαίνεται και στην εικόνα 3.14. Το πρώτο παράδειγμα παλμού που θεωρούμε δίνεται από την σχέση:

$$p(t) = \begin{cases} \cos\left(\frac{\pi}{2T_1}t\right) & ,|t| < T_1 \\ 0 & ,\text{διαφορετικά} \end{cases} \tag{3.34}$$

Στην περίπτωση αυτή είναι εύκολο να δούμε ότι η μέγιστη τιμή του παλμού είναι για $t = 0$ και ισούται με $p_{\max} = 1$, οπότε θα έχουμε και $|p_{\max}|^2 = 1$.

Για να βρούμε θεωρητικά τις τιμές t_1 και t_2 όπου $|p(t_i)|^2 = \frac{1}{2}$, απλά λύνουμε το $\cos(\pi t_i/2/T_1) = \frac{1}{2}$ και βρίσκουμε ότι $t_1 = -T_1/2$ και $t_2 = T_1/2$ οπότε το T_{FWHM} είναι ίσο με $_1$. Όπως βλέπουμε και από τα αποτελέσματα του listing 3.12, η τιμή αυτή ταυτίζεται σχεδόν με την τιμή που υπολογίζεται αριθμητικά από την `fwm_at` και είναι ≈ 0.99.

Το δεύτερο παράδειγμα που θεωρούμε είναι ο παλμός:

$$p(t) = \exp\left(-\frac{t^2}{2T_g^2}\right) \tag{3.35}$$

Ο παραπάνω παλμός ονομάζεται *Gaussian* παλμός. Για να βρούμε το εύρος του, παρατηρούμε ότι και σε αυτή την περίπτωση το μέγιστο $p_{\max} = 1$, βρίσκεται στο $t = 0$ και θέτουμε και πάλι $|p(t_i)|^2 = \frac{1}{2}$ οπότε,

$$\frac{1}{2} = |p(t_i)|^2 = \exp\left(-\frac{t_i^2}{T_g^2}\right) \tag{3.36}$$

Αν πάρουμε το λογάριθμο στην παραπάνω σχέση βρίσκουμε:

$$\frac{t_i^2}{T_g^2} = \ln 2 \tag{3.37}$$

και επομένως

$$t_1 = -T_g\sqrt{\ln 2} \tag{3.38}$$

$$t_2 = T_g\sqrt{\ln 2} \tag{3.39}$$

$$T_{\text{FWHM}} = t_2 - t_1 = 2T_g\sqrt{\ln 2} \tag{3.40}$$

Στον κώδικα του 3.12, θεωρούμε ότι $T_g = \frac{1}{2}$ οπότε θα έχουμε $T_{\text{FWHM}} = 0.83255$ που είναι πολύ κοντά στην τιμή που υπολογίζεται αριθμητικά από τον κώδικα, ≈ 0.83298.

3.9 Το κρουστικό σήμα $\delta(t)$

Μέχρι τώρα είδαμε δύο διαφορετικά σήματα, τον τετραγωνικό παλμό και το φέρον σήμα. Υπάρχει και ένα τρίτο παράδειγμα σήματος που είναι χρήσιμο και καθορίζεται από την συνάρτηση $\delta(t)$. Η $\delta(t)$ ορίζεται μαθηματικά ως η συνάρτηση που έχει την ιδιότητα:

$$\int_{-\infty}^{+\infty} x(t)\delta(t)\mathrm{d}t = x(0) \tag{3.41}$$

Ίσως φαίνεται κάπως περίεργο να ορίζουμε την συνάρτηση με ένα ολοκλήρωμα αλλά θα δούμε ότι η $\delta(t)$ δεν είναι μία συνηθισμένη συνάρτηση. Ας θεωρήσουμε ότι το σήμα $x(t)$ έχει μηδενικές τιμές εκτός ενός διαστήματος $[-,]$, δηλαδή $x(t) = 0$ για $t < -T/2$ ή $t > T/2$. Επομένως το παραπάνω ολοκλήρωμα γράφεται:

$$\int_{-T}^{T} x(t)\delta(t)\mathrm{d}t = x(0) \tag{3.42}$$

Στη συνέχεια χωρίζουμε το διάστημα $[-T,T]$ σε N σημεία t_i τα οποία είναι ισαπέχοντα. Ακολουθώντας την λογική στις σχέσεις (3.5) και (3.7) θέτοντας $t_0 = -T$ και $t_{N-1} = T$, βρίσκουμε ότι:

$$t_i = -T + i\Delta t = -T + i\frac{2T}{N-1} \tag{3.43}$$

Αν θεωρήσουμε ότι το N είναι περιττός αριθμός τότε παρατηρούμε ότι για $p = (N-1)/2$ έχουμε

$$t_p = -T + p\frac{2T}{N-1} = 0 \tag{3.44}$$

Μπορούμε να προσεγγίσουμε το ολοκλήρωμα στην (3.42) με ένα άθροισμα:

$$\int_{-T}^{T} x(t)\delta(t)\mathrm{d}t \approx \Delta t \sum_{n=0}^{N-1} x(t_i)\delta(t_i) \tag{3.45}$$

Χρησιμοποιώντας τις (3.42) και (3.45) καταλήγουμε στην εξής σχέση:

$$\Delta t \sum_{n=0}^{N-1} x(t_i)\delta(t_i) \approx x(0) \tag{3.46}$$

Δεδομένου ότι $t_p = 0$ θα έχουμε $x(0) = x(t_p)$. Επομένως μπορούμε να ικανοποιήσουμε την παραπάνω σχέση αν θέσουμε:

$$\delta(t) = \begin{cases} \frac{1}{\Delta t} & ,-\Delta t/2 \leq t \leq \Delta t/2 \\ 0 & ,\text{διαφορετικά} \end{cases} \tag{3.47}$$

Η κλάση `impulse_signal` προσεγγίζει ένα σήμα $\delta(t)$ σύμφωνα με την (3.47) και φαίνεται στο listing 3.13.

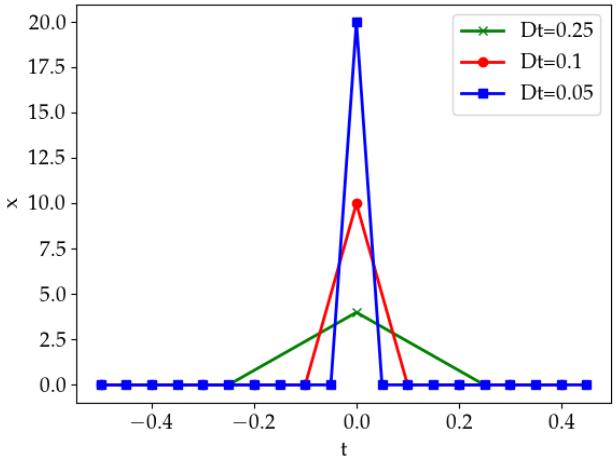
```
118 class impulse_signal(signal):
119     """
120     Impuse signal (delta)
121     """
122     def __init__(self, t, t0 = 0):
123         samples = np.zeros(t.size)
124         Dt = t[1] - t[0]
125         i = np.where( np.abs (t - t0) <= 0.5 * Dt)
126         samples[ i ] = 1 / Dt
127         super().__init__( t = t, samples = samples )
```

Listing 3.13: Η κλάση `impulse_signal`

```
1  import numpy as np
2  import commlib as cl
3  import matplotlib.pyplot as plt
4
5  t1 = np.arange(-0.5, 0.5, 0.1)
6  d1 = cl.impulse_signal(t1)
7
8  t2 = np.arange(-0.5, 0.5, 0.05)
9  d2 = cl.impulse_signal(t2)
10
11 t3 = np.arange(-0.5, 0.5, 0.25)
12 d3 = cl.impulse_signal(t3)
13
14 plt.close('all')
15 plt.figure()
16 d3.plot(line_type = 'g-x')
17 d1.plot(line_type = 'r-o')
18 d2.plot(line_type = 'b-s')
19 plt.legend(['Dt=0.25','Dt=0.1','Dt=0.05'])
```

Listing 3.14: `delta_plot.py`

Στο listing 3.14 δείχνουμε πως κάνουμε την γραφική παράσταση της προσέγγισης της $\delta(t)$ όπως αυτή λαμβάνεται από την κλάση `delta_plot.py`. Στην εικόνα 3.15 δείχνουμε την γραφική παράσταση που προκύπτει. Στην ουσία η προσέγγιση που λαμβάνουμε έχει όλα τα δείγματα ίσα με μηδέν εκτός από το κεντρικό στο $t = 0$, όπου η τιμή ισούται με $1\Delta t$ όπου Δt η διαφορά μεταξύ δύο διαδοχικών σημείων στον άξονα του χρόνου σύμφωνα και με την (3.47). Παρατηρούμε ότι όσο πιο πυκνή είναι η δειγματοληψία στον άξονα του χρόνου δηλαδή



Εικόνα 3.15: Η προσέγγιση του σήματος $\delta(t)$.

όσο μικρότερο είναι το Δt, τόσο μεγαλύτερο είναι το πλάτος του σήματος. Στην (3.45), το άθροισμα προσεγγίζει το ολοκλήρωμα όταν $\Delta t \to 0$, οπότε το πλάτος του σήματος $1/\Delta t$ στο $t = 0$ απειρίζεται και έχουμε $\delta(t) \to \infty$ όταν $t \to 0$. Μπορεί να μας φαίνεται περίεργο αυτό αλλά το σήμα $\delta(t)$ δεν υπάρχει στην πράξη αλλά ένα σήμα που διευκολύνει πολλές στην μαθηματική ανάλυση των συστημάτων επικοινωνιών. Σύμφωνα με την (3.47) αλλά και την εικόνα 3.15 το σήμα $\delta(t)$ είναι ένα σήμα με πολύ μικρή διάρκεια (που τείνει θεωρητικά στο μηδέν) και πολύ μεγάλο στιγμιαίο πλάτος (που τείνει στο άπειρο) μέσα στην διάρκεια αυτή. Θα δούμε στα επόμενα κεφάλαια μερικές περιπτώσεις όπου το σήμα $\delta(t)$ εμφανίζεται στην ανάλυση των επιδόσεων των συστημάτων μας.

3.10 Τι μάθαμε

Στο κεφάλαιο αυτό γνωρίσαμε τα σήματα. Είδαμε δύο βασικά σήματα που θα χρησιμοποιήσουμε στη συνέχεια, τον τετραγωνικό παλμό και το φέρον. Ξεκινήσαμε την υλοποίηση της `commlib` φτιάχνοντας μία κλάση την `signal` που περιγράφει γενικά ένα σήμα και στη συνέχεια υλοποιήσαμε δύο κλάσεις την `carrier` και την `square_pulse` που περιγράφουν τις δύο κατηγορίες σημάτων που αναφέραμε προηγουμένως. Στη συνέχεια είδαμε πως υπολογίζουμε την ενέργεια και την μέση ισχύ ενός σήματος τόσο θεωρητικά όσο και χρησιμοποιώντας την `Python`. Τέλος είδαμε ένα επιπλέον σήμα, το κρουστικό σήμα $\delta(t)$ το οποίο αν και δεν υπάρχει στην πράξη, ωστόσο θα δούμε ότι εμφανίζεται αρκετά συχνά στις αναλύσεις των επιδόσεων των συστημάτων που θα μελετήσουμε.

4. Το πεδίο των συχνοτήτων

4.1 Εισαγωγή

Στο προηγούμενο κεφάλαιο είδαμε τα σήματα στο πεδίο του χρόνου. Στο παρόν κεφάλαιο θα δούμε έναν εναλλακτικό τρόπο να περιγράφουμε τα σήματα στο πεδίο των συχνοτήτων. Πολλές φορές η επίδραση των τηλεπικοινωνιακών συστημάτων είναι πιο εύκολο να περιγραφούν στο πεδίο αυτό. Το πεδίο των συχνοτήτων είναι άμεσα συνυφασμένο με την έννοια του φάσματος που είναι θεμελιώδης τόσο από θεωρητικής αλλά και πρακτικής άποψης - καθώς συχνά το φάσμα αδειοδοτείται και σε κάθε περίπτωση η αποτελεσματική χρήση του μειώνει το κόστος υλοποίησης.

4.2 Αρμονικές συναρτήσεις

Ξεκινάμε με την πιο απλή περίπτωση αναπαράστασης στο πεδίο των συχνοτήτων όπου έχουμε ένα περιοδικό σήμα $x(t)$ με περίοδο T, δηλαδή $x(t) = x(t+T)$. Τότε μπορούμε να γράψουμε το $x(t)$ ως άθροισμα συναρτήσεων $F_m(t)$ κάθε μία εκ των οποίων δίνεται από την σχέση:

$$F_m(t) = \exp\left(j\frac{2\pi m}{T}t\right) \tag{4.1}$$

Στην παραπάνω σχέση θυμίζουμε ότι $j = \sqrt{-1}$ και το m είναι οποιοσδήποτε ακέραιος αριθμός. Σύμφωνα με την ταυτότητα του Euler μπορούμε να γράψουμε $\exp(j\phi) = \cos(\phi) + j\sin(\phi)$ και επομένως

$$F_m(t) = \cos\left(\frac{2\pi m}{T}t\right) + j\sin\left(\frac{2\pi m}{T}t\right) \tag{4.2}$$

Από την παραπάνω σχέση προκύπτει οι αρμονικές συναρτήσεις είναι άθροισμα ενός συνημιτόνου και ενός ημίτονου με κυκλική συχνότητα ω_m:

$$\omega_m = \frac{2\pi m}{T} \tag{4.3}$$

Μπορούμε επίσης να ορίσουμε και την συνήθη συχνότητα f_m,

$$f_m = \frac{\omega_m}{2\pi} = \frac{m}{T} \tag{4.4}$$

οπότε γράφουμε

$$F_m(t) = \exp(j\omega_m t) = \exp(j2\pi f_m t) \tag{4.5}$$

Δεδομένου ότι το $\cos y$ και το $\sin y$ είναι περιοδικές συναρτήσεις με περίοδο 2π, δηλαδή $\cos(\phi + 2\pi n) = \cos\phi$, $\sin(\phi + 2\pi n) = \sin\phi$ προκύπτει ότι και το F_m είναι περιοδική συνάρτηση με περίοδο T και επομένως ισχύει:

$$F_m(t+T) = F_m(t) \tag{4.6}$$

Μπορούμε να υλοποιήσουμε μία κλάση `harmonic` που περιγράφει τα αρμονικά μας σήματα μέσα στην `commlib.py` όπως δείχνει το παρακάτω listing:

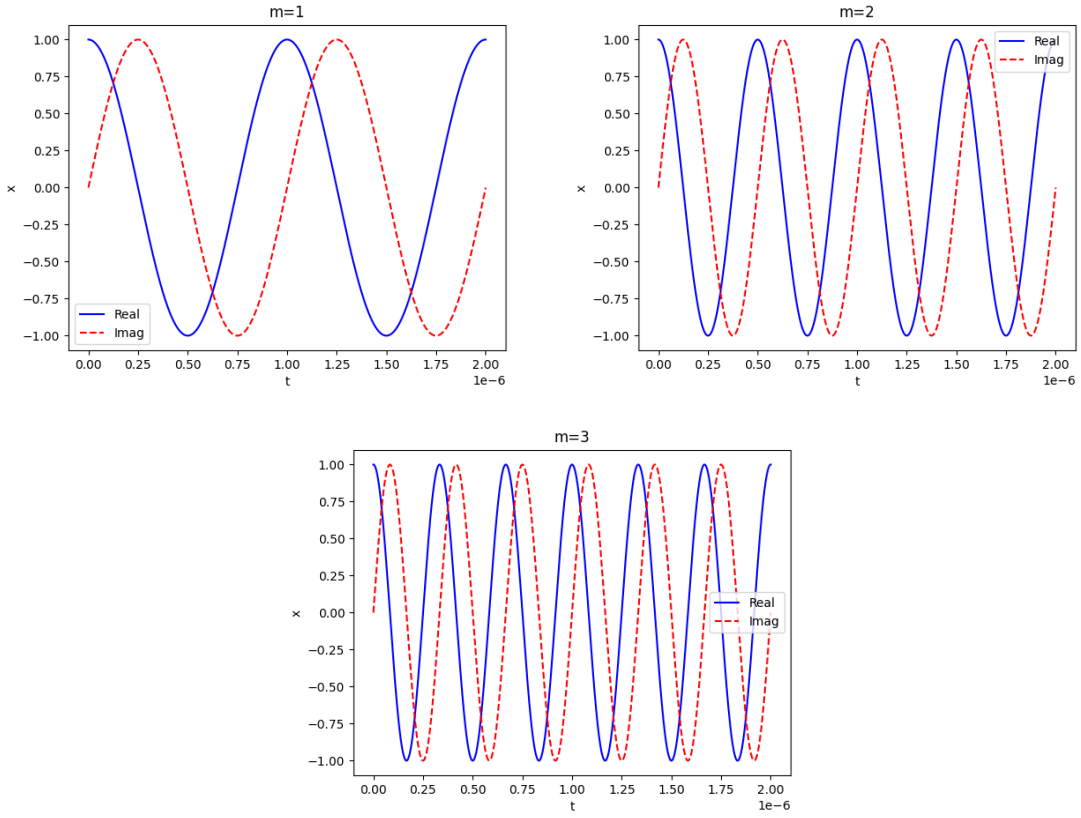
```
109 class harmonic(signal):
110     """
111     Harmonic signal
112     """
113     def __init__(self, t, m, T, C = 1.0):
114         samples = C * np.exp(1j * 2 * np.pi * m / T * t)
115         super().__init__( t = t, samples = samples )
```

Listing 4.1: Η κλάση `harmonic`

Κάνουμε και μία αλλαγή στην κλάση `signal` δεδομένου ότι τα αρμονικά σήματα είναι μιγαδικά. Στην μέθοδο `plot` εισάγουμε και μία ακόμα παράμετρο την `what` που καθορίζει αν κάνουμε plot το πραγματικό μέρος ή το φανταστικό μέρος ή το μέτρο του σήματος[1].

```
1  from commlib import harmonic
2  import matplotlib.pyplot as plt
3  import numpy as np
4
5  T = 1e-6
6  Tmin = 0
7  Tmax = 2 * T
8  N = 1000
9  t = np.linspace(Tmin, Tmax, N)
10
11 plt.close('all')
12 x = harmonic(t, 1, T)
13 x.plot(line_type = 'b-', what = 'real')
14 x.plot(line_type = 'r--', what = 'imag')
15 plt.legend(['Real', 'Imag'])
16 plt.title('m=1')
17
18 plt.figure()
19 x = harmonic(t, 2, T)
20 x.plot(line_type = 'b-', what = 'real')
21 x.plot(line_type = 'r--', what = 'imag')
```

[1] Απλά θυμηθείτε ότι για έναν μιγαδικό αριθμό $z = a + jb$, το πραγματικό μέρος είναι $\Re\{z\} = a$, το φανταστικό μέρος είναι $\Im\{z\} = b$ και το μέτρο του μιγαδικού αριθμού είναι $|z| = \sqrt{a^2 + b^2}$.



Εικόνα 4.1: Οι τρεις πρώτοι αρμονικοί όροι.

```
plt.legend(['Real', 'Imag'])
plt.title('m=2')

plt.figure()
x = harmonic(t, 3, T)
x.plot(line_type = 'b-', what = 'real')
x.plot(line_type = 'r--', what = 'imag')
plt.legend(['Real', 'Imag'])
plt.title('m=3')
```

Listing 4.2: plot_harm.py

Στο listing 4.2 δείχνουμε έναν κώδικα που χρησιμοποιείται για να υπολογίσει και να παραστήσει γραφικά τις αρμονικές συναρτήσεις που προκύπτουν για $m = 1, 2, 3$ και $T = 1\mu\text{s} = 10^{-6}\text{s}$. Στην εικόνα 4.1 δείχνουμε το αποτέλεσμα που παράγεται.

4.3 Σειρά Fourier

Αν θεωρήσουμε ένα σήμα $x(t)$ σε μία διάρκεια $[t_a, t_b]$ όπου τα t_a και t_b είναι πραγματικοί αριθμοί (αλλά κανένας τους δεν είναι ίσος με άπειρο). Για οποιοδήποτε τέτοιο σήμα μπορούμε υπό πολύ γενικές προϋποθέσεις να γράψουμε το $x(t)$ ως ένα άθροισα πάρα πολλών (άπειρων) αρμονικών

συναρτήσεων $F_m(t)$ με $T = t_b - t_a$ που κάθε ένας πολλαπλασιάζεται με έναν συντελεστή C_m.

$$x(t) = \sum_{m=-\infty}^{+\infty} C_m F_m(t) \tag{4.7}$$

Η σειρά στην (4.7) ονομάζεται *σειρά Fourier* του $x(t)$ και τα C_m ονομάζονται *συντελεστές* Fourier του $x(t)$. Τα C_m σχετίζονται με το $x(t)$ ως εξής:

$$C_m = \frac{1}{T}\int_{t_a}^{t_b} x(t)\exp\left(-j\frac{2\pi mt}{T}\right)\mathrm{d}t = \frac{1}{T}\int_{t_a}^{t_b} x(t)\exp(-j2\pi f_m t)\mathrm{d}t \tag{4.8}$$

Παράδειγμα 4.3.1: **Ανάλυση Fourier του τετραγωνικού παλμού**

Για τον τετραγωνικό παλμό

$$p(t) = \begin{cases} 1 & ,-\frac{T_1}{2} \le t \le \frac{T_1}{2} \\ 0 & ,\text{διαφορετικά} \end{cases} \tag{4.9}$$

να βρείτε τους συντελεστές Fourier C_m θεωρώντας ότι $t_a = -\frac{T}{2}$ και $t_b = \frac{T}{2}$

Λύση

Απλά εφαρμόζουμε την (4.8) στην περίπτωση μας. Θα έχουμε [a]

$$C_m = \frac{1}{T}\int_{-T/2}^{T/2} p(t)\exp\left(-j\frac{2\pi mt}{T}\right)\mathrm{d}t =$$
$$\frac{1}{T}\int_{-T_1/2}^{T_1/2}\exp\left(-j\frac{2\pi mt}{T}\right)\mathrm{d}t = \frac{1}{T}\frac{T}{j2\pi m}\left[\exp\left(j\pi\frac{mT_1}{T}\right) - \exp\left(-j\pi\frac{mT_1}{T}\right)\right] =$$
$$\frac{T_1}{T}\frac{T}{\pi mT_1}\sin\left(\frac{\pi mT_1}{T}\right) = \frac{T_1}{T}\mathrm{sinc}\left(\frac{mT_1}{T}\right) \tag{4.10}$$

Στην παραπάνω σχέση χρησιμοποιήσαμε τον ορισμό της συνάρτησης $\mathrm{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$

[a] Θυμηθείτε ότι: $\int_{-b}^{b}\exp(-ax)\mathrm{d}x = \frac{1}{a}\left(\exp(ab) - \exp(-ab)\right)$

4.4 Η σειρά Fourier του τετραγωνικού παλμού

Το παράδειγμα 4.3.1 μας έδειξε ότι η σειρά Fourier του τετραγωνικού σήματος που δίνεται από την (4.9) έχει συντελεστές Fourier που καθορίζονται από την συνάρτηση sinc. Λαμβάνοντας υπόψη την (4.9) μπορούμε να γράψουμε:

$$x(t) = \sum_{m=-\infty}^{+\infty} C_m F_m(t) = \sum_{m=-\infty}^{+\infty} C_m \exp(j2\pi f_m t) \tag{4.11}$$

όπου

$$C_m = \frac{T_1}{T}\text{sinc}(f_m T_1) \tag{4.12}$$

Μπορούμε να υπολογίσουμε τους συντελεστές Fourier χρησιμοποιώντας είτε μαθηματικές εκφράσεις όπως η (4.12) είτε αριθμητικά χρησιμοποιώντας για παράδειγμα τον κανόνα του τραπεζίου. Στην κλάση `signal`, όπου έχουμε να κάνουμε με οποιοδήποτε σήμα μπορούμε να κάνουμε το δεύτερο ενώ στην περίπτωση της κλάσης `square_pulse` κάνουμε το πρώτο.

```
class signal:
    """
    The basic signal class
    """
    def __init__(self, t = None, samples = None):
        """
        Initialize the signal class
        """
        self.t = t
        if self.t is not None:
            self.Dt = t[1] - t[0]
            self.N = t.size
        else:
            self.N = 0
            self.Dt = None

        self.T = np.max(t) - np.min(t)
        self.samples = samples

    def plot(self, line_type = '-o', what = None):
        """
        Plot the signal
        """
        x = self.samples
        if what == 'real':
            x = np.real(x)
        elif what == 'imag':
            x = np.imag(x)
        elif what == 'abs':
            x = np.abs(x)
        plt.plot(self.t, x, line_type)
        plt.xlabel('t')
        plt.ylabel('x')

    def energy(self):
        """
        Calculate signal energy
        """
        return np.trapz(self.samples ** 2.0, self.t)

    def avg_power(self):
        """
        Calcualate average power
        """
        return self.energy() / (np.max(self.t) - np.min(self.t))

    def calc_FS_freqs(self, m):
```

```
        """
        Calculate the frequencies of the Fourier series
        """
        self.fm = m / self.T
        return self.fm

    def calc_FS_coeffs(self, m):
        """
        Calculate the coefficients of the Fourier series
        """
        self.calc_FS_freqs()
        self.Cm = np.zeros(self.fm.size)
        for j, f in enumerate( self.fm ):
            self.Cm[j] = 1 / self.T * np.trapz(
                        self.samples * np.exp(-j*2*np.pi*self.fm[j]*self.t),
                        self.t)

    def calc_FS_sum(self, m):
        """
        Calculate the Fourier series sum
        """
        self.calc_FS_coeffs()
        x = np.zeros( self.N )
        for j, f in enumerate( self.fm ):
            x += self.Cm[j] * self.exp( -j*2*np.pi*self.fm[j] * self.t )

        return x
```

Listing 4.3: Η κλάση `signal`

```
class square_pulse(signal):
    """
    Square pulse
    """
    def __init__(self, t, T1, tcenter = 0.0):
        samples = np.zeros(t.size)
        i = np.where( np.abs (t - tcenter) <= 0.5 * T1)
        samples[ i ] = 1
        self.T1 = T1
        super().__init__( t = t, samples = samples )

    def calc_FS_coeffs(self, m):
        """
        Override the parent function since this is known in closed form
        """
        fm = self.calc_FS_freqs(m)
        return self.T1 / self.T * np.sinc( fm * self.T1 )
```

Listing 4.4: Η κλάση `square_pulse`

Στο listing 4.3 έχουμε προσθέσει μερικές νέες μεθόδους. Καταρχήν η `calc_FS_freqs` υπολογίζει τις συχνότητες που αντιστοιχούν στους ακεραίους `m` που δίνονται ως παράμετρος χρησιμοποιώντας την (4.8). Στη συνέχεια η `calc_FS_coeffs` χρησιμοποιήσει τον κανόνα του τραπεζίου για να υπολογίσουμε τους συντελεστές Fourier C_m. Στην `square_pulse` αντικαθιστούμε αυτή την μέθοδο με μία άλλη που χρησιμοποιεί την μαθηματική σχέση. Τέλος έχουμε και μία μέθοδο την `calc_FS_sum` που υπολογίζει το άθροισμα της σειράς Fourier

σύμφωνα με την (4.11). Στο listing 4.5 δείχνουμε πως μπορούμε να χρησιμοποιήσουμε τις νέες μεθόδους ενώ στην εικόνα 4.2 τα αποτελέσματα που προκύπτουν.

```
import commlib as cl
import numpy as np
import matplotlib.pyplot as plt

m1 = np.arange(-10, 11)
m2 = np.arange(-40, 41)

Tmin = -1
Tmax = 1
T1 = 0.5
N = 1000
t = np.linspace(Tmin, Tmax, N)

x = cl.square_pulse(t, T1)

x1 = x.calc_FS_sum(m1)
x2 = x.calc_FS_sum(m2)

plt.close('all')

plt.figure()
plt.stem(x.fm, np.real(x.Cm) )
plt.xlabel('f')
plt.ylabel('Fourier coefficients')

plt.figure()
plt.plot(t, np.real(x1), label = '21 coeffs')
plt.plot(t, np.real(x2), label = '81 coeffs')
plt.legend()
plt.xlabel('t')
plt.ylabel('Fourier sum')
```

Listing 4.5: fourierseriesexample.py

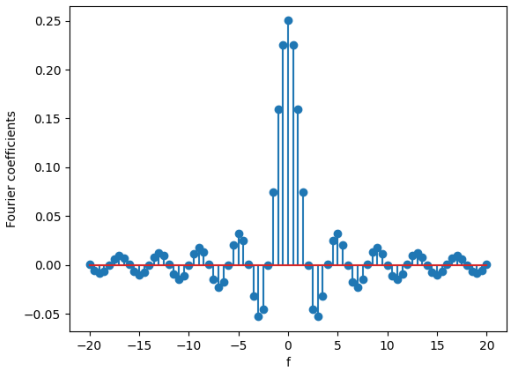
Η αναπαράσταση της σειράς Fourier ισχύει μέσα στο διάστημα ορισμού $[t_a, t_b]$ του σήματος $x(t)$. Τι ισχύει όμως έξω από αυτό το διάστημα; Εφόσον ισχύει η (4.6), μπορούμε να γράψουμε:

$$x(t+T) = \sum_{m=-\infty}^{+\infty} C_m F_m(t+T) = \sum_{m=-\infty}^{+\infty} C_m F_m(t) = x(t) \tag{4.13}$$

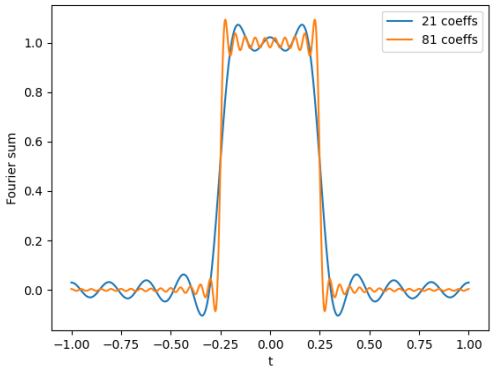
δηλαδή έξω από το πεδίο ορισμού, η σειρά Fourier μας δίνει μία περιοδική επανάληψη του σήματος $x(t)$. Εναλλακτικά θα μπορούσαμε να σκεφτούμε ότι η σειρά Fourier μπορεί να περιγράψει και περιοδικά σήματα $x(t)$ όπου το T είναι η περίοδος. Στην εικόνα 4.3 δείχνουμε την περιοδική επανάληψη που περιγράφει η σειρά Fourier στην περίπτωση του τετραγωνικού παλμού.

4.5 Τι μας δείχνει η σειρά Fourier;

Είναι χρήσιμο να αναφερθούμε στην πληροφορία που μας μεταφέρουν οι συντελεστές C_m. Αν δούμε την (4.11), στην ουσία έχουμε γράψει το σήμα $x(t)$ ως ένα άθροισμα αρμονικών όρων $F_m(t)$ που έχουν συχνότητα f_m. Κάθε αρμονικός όρος $F_m(t)$ σταθμίζεται από τους συντελεστές C_m:

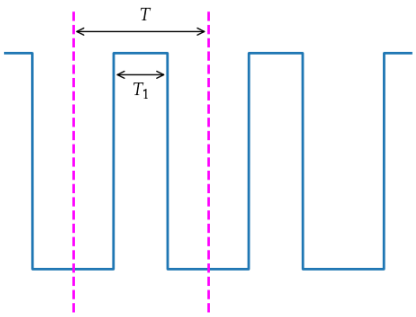


(a)

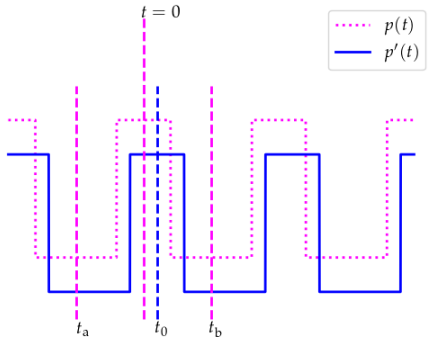


(b)

Εικόνα 4.2: Οι συντελεστές Fourier και το άθροισμα των όρων Fourier



Εικόνα 4.3: Η περιοδική επανάληψη του τετραγωνικού παλμού



Εικόνα 4.4: Μετατόπιση του αρχικού παλμού $p(t)$.

Οι όροι με μεγάλο C_m θα συμμετέχουν περισσότερο στον καθορισμό της συμπεριφοράς του $x(t)$. Επομένως και εφόσον κάθε $F_m(t)$ αντιστοιχεί σε μία συχνότητα f_m, οι συντελεστές Fourier C_m καθορίζουν ποιες συχνότητες f_m είναι σημαντικές για το $x(t)$.

Για παράδειγμα στην περίπτωση του τετραγωνικού παλμού, τα C_m δίνονται από την (4.12) και έχουν παρασταθεί γραφικά στην Εικόνα 4.2. Γίνεται φανερό ότι οι συντελεστές C_m κοντά στις χαμηλές συχνότητες ($f_m \cong 0$) είναι κατά κανόνα πιο ισχυροί από τους συντελεστές σε μεγαλύτερες συχνότητες. Αυτό σημαίνει ότι οι χαμηλές συχνότητες καθορίζουν σε μεγαλό βαθμό την συμπεριφορά του σήματος. Συχνά αναφερόμαστε στην γραφική παράσταση του C_m με την συχνότητα f_m ως το *φάσμα* του σήματος.

Θα πρέπει να παρατηρήσουμε ότι εν γένει οι συντελεστές Fourier είναι *μιγαδικοί* αριθμοί. Για παράδειγμα τι θα γινόταν αν μετατοπίζαμε τον περιοδικό τετραγωνικό παλμό κατά t_0 όπως δείχνει η εικόνα 4.4;

Ο νέος παλμός που προκύπτει $p'(t)$ θα έχει συντελεστές Fourier:

$$D_m = \frac{1}{T}\int_{t_a}^{t_b} p'(t)\exp\left(-j\frac{2\pi mt}{T}\right)\mathrm{d}t \tag{4.14}$$

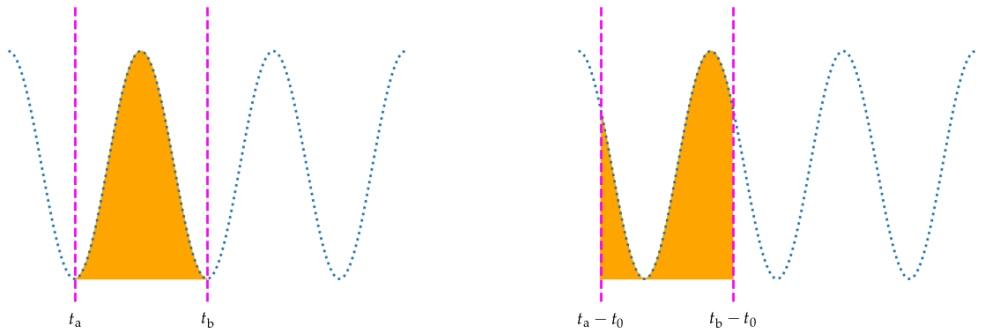
Όπως δείχνει η εικόνα 4.4 το $p'(t)$ είναι μία μετατοπισμένη έκδοση του $p(t)$ κατά t_0. Αν ο αρχικός παλμός $p(t)$ έχει κέντρο το $t = 0$ τότε ο τελικός παλμός $p'(t)$ θα έχει κέντρο το $t = t_0$, οπότε $p'(t) = p(t - t_0)$. Οπότε:

$$D_m = \frac{1}{T}\int_{t_a}^{t_b} p(t-t_0)\exp\left(-j\frac{2\pi mt}{T}\right)\mathrm{d}t \tag{4.15}$$

Αν θέσουμε $\tau = t - t_0$ θα έχουμε:

$$D_m = \frac{1}{T}\exp\left(-j\frac{2\pi mt_0}{T}\right)\int_{t_a-t_0}^{t_b-t_0} p(\tau)\exp\left(-j\frac{2\pi m\tau}{T}\right)\mathrm{d}\tau \tag{4.16}$$

Το ολοκλήρωμα στην (4.16) έχει ως όρισμα την συνάρτηση $p(\tau)\exp\left(-j\frac{2\pi m\tau}{T}\right)$ η οποία είναι περιοδική ως προς τ με περίοδο T. Το ολοκλήρωμα μίας περιοδικής συνάρτησης μέσα σε μία



Εικόνα 4.5: Το εμβαδόν μία περιοδικής συνάρτησης μέσα σε μία ολόκληρη περίοδο παραμένει το ίδιο ακόμα και αν μετατοπίσουμε τα άκρα της περιόδου.

περίοδο είναι το ίδιο ακόμα και αν μετατοπίσουμε τα όρια του κατά $-t_0$ όπως δείχνει και η εικόνα 4.5.

Μπορούμε επομένως να γράψουμε:

$$D_m = \frac{1}{T}\exp\left(-j\frac{2\pi m t_0}{T}\right)\int_{t_a}^{t_b} p(\tau)\exp\left(-j\frac{2\pi m\tau}{T}\right)\mathrm{d}\tau = C_m\exp\left(-j2\pi f_m t_0\right) \tag{4.17}$$

Παρατηρούμε ότι σε σχέση με την (4.12) έχει προστεθεί ένας μιγαδικός όρος $\exp\left(-j2\pi f_m t_0\right)$ και επομένως οι συντελεστές Fourier D_m είναι *μιγαδικοί* αριθμοί.

Στο listing 4.6 δείχνουμε μία γενίκευση της κλάσης `square_pulse` για την οποία η μέθοδος `calc_FS_coeffs` λαμβάνει υπόψη της και το κέντρο του παλμού στην παράμετρο `tcenter`.

```
class square_pulse(signal):
    """
    Square pulse
    """
    def __init__(self, t, T1, tcenter = 0.0):
        self.tcenter = tcenter
        samples = np.zeros(t.size)
        i = np.where( np.abs (t - tcenter) <= 0.5 * T1)
        samples[ i ] = 1
        self.T1 = T1
        super().__init__( t = t, samples = samples )

    def calc_FS_coeffs(self, m):
        """
        Override the parent function since this is known in closed form
        """
        fm = self.calc_FS_freqs(m)
        self.Cm = self.T1 / self.T * np.sinc( fm * self.T1 ) * np.exp( -1j * 2*
    np.pi * fm * self.tcenter )
        return self.Cm
```

Listing 4.6: Μία νέα έκδοση της `square_pulse`

```
import commlib as cl
import numpy as np
import matplotlib.pyplot as plt

m1 = np.arange(-11,12)
m2 = np.arange(-21,22)
Tmin = -1
Tmax = 1
T1 = 0.25
N = 1000
t = np.linspace(Tmin, Tmax, N)

x = cl.square_pulse(t, T1, tcenter = T1/2)

x1 = x.calc_FS_sum(m1)
x2 = x.calc_FS_sum(m2)

plt.close('all')

plt.figure()
plt.plot(x.fm, np.real(x.Cm),'-o',label = 'Re' )
plt.plot(x.fm, np.imag(x.Cm),'-rs',label = 'Im' )
plt.xlabel('f')
plt.ylabel('Fourier coefficients')
plt.legend()

plt.figure()
plt.plot(t, np.real(x1), label = '21 coeffs')
plt.plot(t, np.real(x2), label = '41 coeffs')
plt.plot(t, x.samples, '--')
plt.legend()
plt.xlabel('t')
plt.ylabel('Fourier sum')
```

Listing 4.7: Εφαρμογή της νέας έκδοσης της textttsquare_pulse

Στο listing 4.7 δείχνουμε πως χρησιμοποιούμε την νέα μορφή της κλάσης. Στην Εικόνα **??** δείχνουμε το πραγματικό και το φανταστικό μέρος των συντελεστών Fourier D_m.

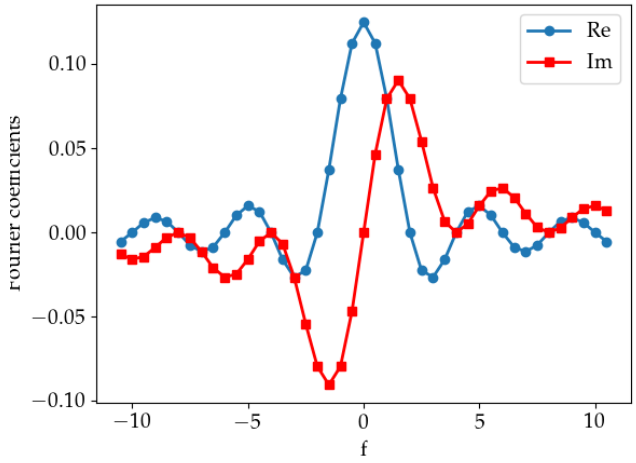
Πως γίνεται να ξεκινούμε από ένα πραγματικό σήμα όπως το $p'(t)$ και να έχουμε μιγαδικούς συντελεστές Fourier; Όταν τους αθροίζουμε, το αποτέλεσμα δεν θα βγαίνει και αυτό μιγαδικό; Αν κοιτάξουμε προσεκτικά τους συντελεστές Fourier στην εικόνα 4.16 θα δούμε ότι το πραγματικό μέρος (μπλε γραμμή) είναι συμμετρικό με το f_m ενώ το φανταστικό μέρος είναι αντίσυμμετρικό. Δηλαδή ισχύει

$$\Re\{D_m\} = \Re\{D_{-m}\} \tag{4.18a}$$

$$\Im\{D_m\} = -\Im\{D_{-m}\} \tag{4.18b}$$

Αυτό σημαίνει ότι το D_{-m} είναι το μιγαδικό συζυγές του D_m, δηλαδή ισχύει:

$$(D_{-m})^* = D_m \tag{4.19}$$



όπου με z^* έχουμε παραστήσει το μιγαδικό συζυγές του z. Για $m = 0$ προκύπτει ότι το D_0 θα είναι ίδιο με το $(D_0)^*$ και επομένως το D_0 θα πρέπει να είναι πραγματικό. Η σειρά Fourier επομένως μπορεί να γραφεί

$$\sum_{m=-\infty}^{+\infty} D_m F_m(t) = \sum_{m=1}^{+\infty} D_m F_m(t) + \sum_{m=1}^{+\infty} (D_m)^* F_m^*(t) + D_0 F_0(t) \tag{4.20}$$

όπου απλά $F_0(t) = \exp(j0) = 1$. Η παραπάνω σχέση δείχνει για κάθε όρος του αθροίσματος Fourier $D_m F_m(t)$ αντιστοιχεί και ένας όρος που είναι ο μιγαδικός συζυγής του και επομένως τα φανταστικά μέρη αναιρούνται με αποτέλεσμα το τελικό άθροισμα να είναι πραγματικός αριθμός.

Ένας τρόπος για να αντιληφθούμε καλύτερα την επίδραση του φανταστικού μέρους των συντελεστών Fourier C_m είναι να χρησιμοποιήσουμε την λεγόμενη πολική μορφή των C_m. Κάθε μιγαδικός αριθμός $z = a + jb$ γράφεται ως $z = |z|\mathrm{e}^{j\phi}$ όπου $z = \sqrt{a^2 + b^2}$ είναι το μέτρο του z και η ϕ είναι η φάση του που καθορίζεται από τις σχέσεις:

$$\cos\phi = \frac{a}{|z|} \tag{4.21a}$$

$$\sin\phi = \frac{b}{|z|} \tag{4.21b}$$

Επομένως μπορούμε να γράψουμε:

$$C_m = |C_m|\mathrm{e}^{j\phi_m} \tag{4.22}$$

Σύμφωνα με την (4.11) θα έχουμε:

$$x(t) = \sum_{m=-\infty}^{\infty} |C_m| \exp(j2\pi f_m t + j\phi_m) \tag{4.23}$$

Η παραπάνω σχέση δηλώνει ότι η φάση ϕ_m που έχει ο συντελεστής Fourier καθορίζει και την διαφορά φάσης που έχει ο κάθε όρος της σειράς Fourier σε σχέση με την συχνότητα.

4.6 Μετασχηματισμός Fourier

Τι συμβαίνει όταν δεν έχουμε σήματα ορισμένα σε μία περιορισμένη διάρκεια $[t_a, t_b]$ αλλά σε όλο τον άξονα του t από $-\infty$ έως $+\infty$; Στην περίπτωση δεν μπορούμε να χρησιμοποιήσουμε την σειρά Fourier εφόσον $T \to +\infty$, αλλά τον *μετασχηματισμό* Fourier. Ο μετασχηματισμός Fourier $\mathscr{F}$, ορίζεται ως:

$$X(f) = \mathscr{F}\{x(t)\} = \int_{-\infty}^{+\infty} x(t) \mathrm{e}^{-j2\pi ft} \mathrm{d}t \tag{4.24}$$

Μπορούμε επίσης να υπολογίσουμε το σήμα $x(t)$ στο πεδίο του χρόνου χρησιμοποιώντας τον αντίστροφο μετασχηματισμό Fourier $\mathscr{F}^{-1}$,

$$x(t) = \mathscr{F}^{-1}\{X(f)\} \int_{-\infty}^{+\infty} X(f) \mathrm{e}^{j2\pi ft} \mathrm{d}f \tag{4.25}$$

Υπό μία έννοια, ο μετασχηματισμός Fourier είναι κάτι αντίστοιχο με την σειρά Fourier. Το ολοκλήρωμα της (4.25) είναι στην ουσία ένα άθροισμα όρων $X(f_i)\exp(j2\pi f_i t)\Delta f$ παρόμοιο με το άθροισμα στην (4.11).

Παράδειγμα 4.6.1: Μετασχηματισμός Fourier του τετραγωνικού παλμού

Για τον τετραγωνικό παλμό

$$p(t) = \begin{cases} 1 & , -\frac{T_1}{2} \le t \le \frac{T_1}{2} \\ 0 & , \text{διαφορετικά} \end{cases} \tag{4.26}$$

να βρείτε το μετασχηματισμό Fourier $P(f)$.

Λύση

Χρησιμοποιούμε την (4.24) και έχουμε

$$P(f) = \int_{-\infty}^{+\infty} p(t) \mathrm{e}^{-j2\pi ft} \mathrm{d}t = \int_{-T_1/2}^{+T_1/2} \mathrm{e}^{-j2\pi ft} \mathrm{d}t = \frac{\mathrm{e}^{-j\pi fT_1} - \mathrm{e}^{j\pi fT_1}}{-j2\pi f} = T_1 \frac{\sin(\pi fT_1)}{\pi fT_1} = T_1 \mathrm{sinc}(fT_1) \tag{4.27}$$

Στην παραπάνω σχέση χρησιμοποιήσαμε τον ορισμό της συνάρτησης $\mathrm{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$ και παρατηρούμε ότι ο μετασχηματισμός Fourier $P(f)$ μοιάζει ως συνάρτηση με τους συντελεστές Fourier (4.12).

4.6.1 Πως υπολογίζεται με υπολογιστή ο μετασχηματισμός Fourier

Χρειαζόμαστε καταρχήν έναν αποτελεσματικό τρόπο για να υπολογίζουμε τον μετασχηματισμό Fourier για τα σήματα μας. Θα μπορούσαμε να χρησιμοποιήσουμε αριθμητική ολοκλήρωση

τόσο για την (4.24) όσο και για την (4.25). Ωστόσο δεδομένου ότι θα πρέπει να υπολογίσουμε το αποτέλεσμα για πολλές συχνότητες f ή πολλές χρονικές στιγμές t καταλαβαίνουμε ότι ο χρόνος υπολογισμού ενδέχεται να είναι πολύ μεγάλος.

Ας περιοριστούμε σε συγκεκριμένες N συχνότητες f_k μέσα σε ένα διάστημα $[-F_{\max}, F_{\max}]$ που δίνονται από την σχέση $f_k = -F_{\max} + k\Delta f$ με $\Delta f = 2T_{\max}/N$. Επίσης θεωρούμε ότι το σήμα $x(t)$ έχει αμελητέες τιμές εκτός ενός μεγάλου διαστήματος $[-T_{\max}, T_{\max}]$. Μέσα στο διάστημα αυτό θεωρούμε N χρονικές στιγμές $t_n = -T_{\max} + n\Delta t$ όπου $\Delta t = 2T_{\max}/N$. Επιλέγουμε την παρακάτω σχέση να συνδέει τα Δf και Δt:

$$\Delta t \Delta f = \frac{1}{N} \tag{4.28}$$

Αν ισχύει η παραπάνω σχέση θα έχουμε

$$F_{\max} T_{\max} = (N/2)^2 \Delta f \Delta t = N/4 \tag{4.29a}$$

$$F_{\max} \Delta t = (N/2)\Delta f \Delta t = 1/2 \tag{4.29b}$$

$$T_{\max} \Delta f = (N/2)\Delta t \Delta f = 1/2 \tag{4.29c}$$

Στην περίπτωση αυτή θα έχουμε

$$f_k t_n = (-F_{\max} + k\Delta f)(-T_{\max} + n\Delta t) = F_{\max} T_{\max} - F_{\max} n\Delta t - T_{\max} k\Delta f + kn\Delta t \Delta f = N/4 - k/2 - n/2 + kn/N \tag{4.30}$$

Μπορούμε να προσεγγίσουμε το ολοκλήρωμα στην (4.24) με το εξής άθροισμα,

$$X(f_k) \approx \Delta t \sum_{n=0}^{N-1} x(t_n) \exp(-j2\pi f_k t_n) = \Delta t \sum_{n=0}^{N-1} x(t_n) \exp(-j\pi N/2 + j\pi k + j\pi n + j2\pi kn/N) \tag{4.31}$$

Αν γράψουμε λίγο διαφορετικά την παραπάνω σχέση θα έχουμε:

$$X(f_k) \mathrm{e}^{-j\pi k} = \Delta t \mathrm{e}^{-j\pi N/2} \sum_{n=0}^{N-1} x(t_n) \mathrm{e}^{j\pi n} \exp(-j2\pi kn/N) \tag{4.32}$$

Στην παραπάνω σχέση το άθροισμα που εμφανίζεται περιγράφει έναν διακριτό μετασχηματισμό Fourier (Discrete Fourier Transform - DFT). Γενικότερα εάν έχουμε δείγματα y_n τότε ο DFT του y_n ορίζεται ως:

$$Y_k = \mathrm{DFT}\{y_n\} = \sum_{n=0}^{N-1} y_n \exp(-j2\pi kn/N) \tag{4.33}$$

Παρατηρούμε επομένως ότι αν έχουμε ένα γρήγορο τρόπο να υπολογίζουμε τον DFT μπορούμε να τον χρησιμοποιήσουμε για να προσεγγίσουμε και τον μετασχηματισμό Fourier (4.24). Στη συνέχεια θα δούμε πως υπολογίζεται ο DFT στην Python. Μπορούμε να αντιστρέψουμε τον DFT και να υπολογίσουμε τα y_n από τα Y_k. Αποδεικνύεται ότι ο αντίστροφος μετασχηματισμός DFT (inverse DFT - IDFT) δίνεται από την σχέση:

$$y_n = \text{IDFT}\{Y_k\} = \frac{1}{N}\sum_{k=0}^{N-1} Y_k \exp(j2\pi kn/N) \tag{4.34}$$

4.6.2 Υπολογισμός του DFT

Στη βιβλιοθήκη numpy ο DFT υπολογίζεται με την βοήθεια του γρήγορου μετασχηματισμού Fourier (fast Fourier transform - FFT). Ο FFT είναι ένας πολύ αποτελεσματικός τρόπος να υπολογίζεται ο DFT.

Για να δούμε τον υπολογισμό μέσω του FFT στην πράξη θα κάνουμε αρχικά μία μικρή μετατροπή στην commlib.py για να τυπώνουμε με εύχρηστο τρόπο τα στοιχεία ενός numpy array. Στο listing 4.8 δείχνουμε την συνάρτηση print_array η οποία τυπώνει με ακρίβεια δύο δεκαδικών ψηφίων τα στοιχεία ενός numpy πίνακα τα οποία μπορεί να είναι και μιγαδικοί αριθμοί.

```
def print_array(x):
    for i, xx in enumerate(x):
        st = str(i) + ' : ' + '%2.2f' %np.real(xx)
        if np.imag(xx) > 0:
            st+= '+i%2.2f' %np.imag(xx)
        elif np.imag(xx) < 0:
            st+= '-i%2.2f' %np.imag(-xx)
        print(st)
```

Listing 4.8: Η συνάρτηση print_array

```
import numpy as np
from commlib import print_array

N = 4
k = np.arange(0, N, dtype=int)
np.random.seed(1)
x = np.random.rand(N)
X = np.fft.fft(x)

print('original signal:')
print_array(x)
print('original DFT:')
print_array(X)

x1 = np.roll(x, 1)
X1 = np.fft.fft(x1)
X1b = X * np.exp(-1j*2*np.pi/N*k)

print('shifted signal:')
print_array(x1)
print('DFT of shifted signal (FFT):')
print_array(X1)
print('DFT of shifted signal (theory):')
```

```
original signal:
0 : 0.42
1 : 0.72
2 : 0.00
3 : 0.30
original DFT:
0 : 1.44
1 : 0.42-i0.42
2 : -0.61
3 : 0.42+i0.42
```

Εικόνα 4.6: Ο αρχικός υπολογισμός του DFT

```
shifted signal:
0 : 0.30
1 : 0.42
2 : 0.72
3 : 0.00
DFT of shifted signal (FFT):
0 : 1.44
1 : -0.42-i0.42
2 : 0.61
3 : -0.42+i0.42
DFT of shifted signal (theory):
0 : 1.44
1 : -0.42-i0.42
2 : 0.61+i0.00
3 : -0.42+i0.42
```

Εικόνα 4.7: Ο DFT του σήματος που προκύπτει από την κυκλική μετατόπιση μίας θέσης

```
print_array(X1b)

y = np.fft.fftshift(x)
Y = np.fft.fft(y)
Yb = X * np.exp(-1j*np.pi*k)
print('fftshift:')
print_array(y)
print('DFT of fftshifted signal:')
print_array(Y)
print('DFT of fftshifted signal (theory):')
print_array(Yb)
```

Listing 4.9: Το αρχείο `tst_dft.py`

Στη συνέχεια χρησιμοποιούμε το listing 4.9 για να υπολογίσουμε τον DFT σε μία απλή περίπτωση όπου το σήμα y_k αποτελείται από τέσσερις τιμές ($N = 4$) οι οποίες επιλέγονται τυχαία με την βοήθεια της `numpy.random.rand`. Προσέξτε ότι πριν την κλήση της `numpy.random.rand` χρησιμοποιούμε την `numpy.random.seed` για να αρχικοποιήσουμε την γεννήτρια τυχαίων αριθμών της `numpy` στην ίδια αρχική κατάσταση δηλαδή την 1. Ο υπολογισμός του DFT γίνεται με την συνάρτηση `numpy.fft.fft`

```
X = np.fft.fft(x)
```

Στη συνέχεια χρησιμοποιούμε την `print_array` για να τυπώσουμε τα αποτελέσματα των υπολογισμών. Στην εικόνα 4.6 δείχνουμε τον υπολογισμό του αρχικού DFT με τον FFT. Ο αρχικός μας πίνακας x_n έχει τιμές $[0.42, 0.72, 0.00, 0.30]$ και ο DFT του X_k είναι $[1.44, 0.42 - j0.42, -0.61, 0.42 + j0.42]$.

Στη συνέχεια δείχνουμε την συμβαίνει όταν κάνουμε κυκλική ολίσθηση του x_n προς τα δεξιά κατά μία θέση. Στην περίπτωση αυτή όλα τα στοιχεία του x_n ολισθαίνουν μία θέση δεξιά και το τελευταίο στοιχείο έρχεται στην αρχή. Έτσι ο $[0.42, 0.72, 0.00, 0.30]$ γίνεται

```
fftshift:
0 : 0.00
1 : 0.30
2 : 0.42
3 : 0.72
DFT of fftshifted signal:
0 : 1.44
1 : -0.42+i0.42
2 : -0.61
3 : -0.42-i0.42
DFT of fftshifted signal (theory):
0 : 1.44
1 : -0.42+i0.42
2 : -0.61-i0.00
3 : -0.42-i0.42
```

Εικόνα 4.8: Ο DFT του σήματος που προκύπτει από την κυκλική μετατόπιση δύο θέσεων

$[0.30, 0.42, 0.72, 0.00]$. Υπάρχει μία βασική ιδιότητα του DFT που λέει ότι αν ο αρχικός πίνακας x_n ολισθήσει κυκλικά κατά p θέσεις προς τα δεξιά και προκύψει ο πίνακας z_n τότε ο DFT Z_k του z_n συνδέεται με τον DFT X_k του x_n ως εξής:

$$Z_k = X_k \exp\left(-\frac{j2\pi pk}{N}\right) \tag{4.35}$$

Στην εικόνα 4.7 δείχνουμε πως υπολογίζεται ο DFT του πίνακα που έχει προκύψει από την κυκλική ολίσθηση κατά μία θέση ($p = 1$) και συγκρίνουμε τα αποτελέσματα όταν εφαρμόζουμε τον FFT απευθείας στο ολισθημένο πίνακα και όταν εφαρμόζουμε την (4.35). Και στις δύο περιπτώσεις το αποτέλεσμα για το Z_k είναι το ίδιο δηλαδή $[1.44, -0.42 - j0.42, 0.61, -0.42 + j0.42]$.

Ας υποθέσουμε τώρα ότι εφαρμόζουμε την κυκλική ολίσθησε για $p = N/2$, δηλαδή ολισθαίνουμε τα στοιχεία του πίνακα κατά τις μισές θέσεις. Τότε η (4.35) θα γραφεί:

$$Z_k = X_k \mathrm{e}^{-j\pi k} \tag{4.36}$$

Θα δούμε στη συνέχεια ότι αυτή η περίπτωση $p = N/2$ είναι σημαντική και η `numpy` έχει μία ειδική συνάρτηση την `numpy.fft.fftshift` ειδικά για αυτήν. Στην περίπτωση του σήματος που εξετάζαμε στην εικόνα 4.6 θα έχουμε $N/2 = 2$ οπότε θα έχουμε ολίσθηση δύο θέσεων προς τα δεξιά. Στην εικόνα 4.8, δείχνουμε και πάλι το αποτέλεσμα που λαμβάνουμε όταν εφαρμόζουμε την `numpy.fft.fftshift` και τον FFT καθώς και την (4.36). Και στις δύο περιπτώσεις το αποτέλεσμα είναι το ίδιο, δηλαδή $[1.44, -0.42 + j0.42, -0.61, -0.42 - j0.42]$.

4.6.3 Υπολογισμός του IDFT

Δεδομένου ότι η (4.33) που περιγράφει τον DFT και η (4.34) που περιγράφει τον αντίστροφο του μοιάζουν αρκετά δεν πρέπει να αποτελεί έκπληξη ότι η `numpy` διαθέτει και έναν τρόπο για τον αποδοτικό υπολογισμό του IDFT που ονομάζεται IFFT. Ο συγκεκριμένος υπολογισμός πραγματοποιείται από την συνάρτηση `numpy.fft.ifft`. Έχει ενδιαφέρον να επαναλάβουμε τους αντίστοιχους υπολογισμούς που κάναμε στην προηγούμενη υποενότητα. Στην περίπτωση αυτή αν ολισθήσουμε κυκλικά τον πίνακα X_k κατά p θέσεις δεξιά και πάρουμε τον πίνακα Z_k τότε έχουμε μία σχέση ανάλογη με την (4.35) με την διαφορά στο πρόσημο μέσα στο εκθετικό,

$$z_n = x_n \exp\left(\frac{j2\pi pn}{N}\right) \tag{4.37}$$

όπου το z_n είναι ο IDFT του Z_k. Και πάλι η ειδική περίπτωση $p = N/2$ παρουσιάζει ενδιαφέρον όπως θα δούμε και στη συνέχεια. Για $p = N/2$ έχουμε:

$$z_n = x_n e^{j\pi n} \tag{4.38}$$

```
import numpy as np
from commlib import print_array

N = 4
n = np.arange(0, N, dtype=int)
np.random.seed(1)
x = np.random.rand(N)
X = np.fft.fft(x)
xi = np.fft.ifft(X)

print('original signal:')
print_array(x)
print('original DFT:')
print_array(X)
print('inverted DFT:')
print_array(xi)

X1 = np.roll(X, 1)
x1 = np.fft.ifft(X1)
x1b = x * np.exp(1j*2*np.pi/N*n)

print('shifted spectrum:')
print_array(X1)
print('IDFT of shifted spectrum (IFFT):')
print_array(x1)
print('IDFT of shifted spectrum (theory):')
print_array(x1b)

Y = np.fft.fftshift(X)
y = np.fft.ifft(Y)
yb = x * np.exp(1j*np.pi*n)
print('fftshift:')
print_array(Y)
print('IDFT of fftshifted spectrum:')
print_array(y)
print('DFT of fftshifted spectrum (theory):')
print_array(yb)
```

Listing 4.10: Το αρχείο `tst_idft.py`

Στο listing 4.10 δείχνουμε πως χρησιμοποιούμε την `numpy.fft.ifft` για να αντιστρέψουμε τον DFT που υπολογίσαμε στο 4.10. Οι γραμμές

```
X = np.fft.fft(x)
xi = np.fft.ifft(X)
```

στην ουσία υπολογίζουν τον DFT του αρχικού μας σήματος και μετά υπολογίζουν τον IDFT ο οποίος θα πρέπει να είναι ο ίδιος με το αρχικό σήμα αφού ο IDFT αντιστρέφει τον DFT. Αυτό φαίνεται και στην εικόνα 4.9 όπου φαίνεται ότι το αποτέλεσμα του IDFT είναι το αρχικό σήμα. Στην εικόνα 4.10, δείχνουμε τη σύγκριση των αποτελεσμάτων όταν κάνουμε κυκλική ολίσθηση του X_k κατά μία θέση δεξιά και εφαρμόσουμε τον IDFT καθώς και το θεωρητικό αποτέλεσμα που

```
original signal:
0 : 0.42
1 : 0.72
2 : 0.00
3 : 0.30
original DFT:
0 : 1.44
1 : 0.42-i0.42
2 : -0.61
3 : 0.42+i0.42
inverted DFT:
0 : 0.42
1 : 0.72
2 : 0.00
3 : 0.30
```

Εικόνα 4.9: DFT και IDFT

```
shifted spectrum:
0 : 0.42+i0.42
1 : 1.44
2 : 0.42-i0.42
3 : -0.61
IDFT of shifted spectrum (IFFT):
0 : 0.42
1 : 0.00+i0.72
2 : -0.00
3 : 0.00-i0.30
IDFT of shifted spectrum (theory):
0 : 0.42
1 : 0.00+i0.72
2 : -0.00+i0.00
3 : -0.00-i0.30
```

Εικόνα 4.10: Ο IDFT που προκύπτει από την κυκλική μετατόπιση μίας θέσης

περιμένουμε από την (4.38). Τέλος στην εικόνα 4.11 δείχνουμε τι συμβαίνει όταν κάνουμε κυκλική ολίσθηση $p = N/2 = 2$ θέσεων του X_k.

4.6.4 Χρήση του DFT για τον υπολογισμό του μετασχηματισμού Fourier

Όπως αναφέραμε και παραπάνω, στόχος μας είναι να υπολογίσουμε τον μετασχηματισμό Fourier που δίνεται από την (4.24) μέσω του DFT. Ας ξαναγράψουμε την (4.32) εδώ για ευκολία θεωρώντας ότι το πλήθος των σημείων N είναι πολλαπλάσιο του 4 οπότε υπάρχει ακέραιος q τέτοιος ώστε $N = 4q$ οπότε και $\mathrm{e}^{-j\pi N/2} = \mathrm{e}^{-j2\pi q} = 1$.

$$\tilde{X}_k \mathrm{e}^{-j\pi k} = \Delta t \mathrm{DFT}\left\{x_n \mathrm{e}^{j\pi n}\right\} \tag{4.39}$$

```
fftshift:
0 : -0.61
1 : 0.42+i0.42
2 : 1.44
3 : 0.42-i0.42
IDFT of fftshifted spectrum:
0 : 0.42
1 : -0.72
2 : 0.00
3 : -0.30
DFT of fftshifted spectrum (theory):
0 : 0.42
1 : -0.72+i0.00
2 : 0.00-i0.00
3 : -0.30+i0.00
```

Εικόνα 4.11: Ο IDFT που προκύπτει από την κυκλική μετατόπιση δύο θέσεων

Στην παραπάνω εξίσωση έχουμε θέσει $\tilde{X}_k = X(f_k)$ και $x_n = x(t_n)$ ενώ έχουμε χρησιμοποιήσει και τον ορισμό του DFT στην (4.33). Ας ορίσουμε τώρα και τον μετασχηματισμό SHIFT που στην ουσία περιγράφει το τι κάνει η κυκλική μετατόπιση προς τα δεξιά κατά $p = N/2$ θέσεις:

$$\text{SHIFT}\{z_n\} = \begin{cases} z_{k-N/2} & \text{εάν } n \geq N/2, \\ z_{k+N/2} & \text{εάν } 0 \leq n < N/2 \end{cases} \tag{4.40}$$

Αν προσέξουμε την (4.40), θα δούμε ότι φέρνει τα στοιχεία του z_n που είναι από την μέση και μετά ($k \leq N/2$) στην αρχή του τελικού πίνακα και τα στοιχεία του z_n που είναι στην αρχή ($0 \leq \ n < N/2$) μετατοπίζονται $/2$ δεξιά, όπως ακριβώς δηλαδή συμπεριφέρεται η `numpy.fft.fftshift`.

Υπάρχουν μερικές χρήσιμες ιδιότητες που διέπουν την SHIFT. Είναι φανερό ότι όταν εφαρμόσουμε δύο φορές της SHIFT τότε θα πάρουμε το αρχικό πίνακα, δηλαδή:

$$\text{SHIFT}\{\text{SHIFT}\{z_n\}\} = z_n \tag{4.41}$$

Επίσης όταν έχουμε γινόμενο δύο πινάκων το SHIFT του γινομένου θα είναι το γινόμενο των επιμέρους SHIFT,

$$\text{SHIFT}\{z_n w_n\} = \text{SHIFT}\{z_n\}\,\text{SHIFT}\{w_n\} \tag{4.42}$$

Στην περίπτωση όπου $z_n = e^{j\pi n}$ τότε οι τιμές του πίνακα z_n θα είναι $[1, -1, 1, -1, \cdots]$ δηλαδή επαναλαμβάνονται ως προς το n με περίοδο 2. Είναι επομένως προφανές ότι αν $N = 4q$ οπότε και $N/2 = 2q$ το SHIFT δεν επιδρά καθόλου στο σήμα δηλαδή $\text{SHIFT}\{z_n\} = z_n$.

Χρησιμοποιώντας τις ιδιότητες που είδαμε παραπάνω και την (4.38) μπορούμε να γράψουμε:¨

$$\tilde{X}_k \mathrm{e}^{-j\pi k} = \Delta t \text{DFT}\left\{x_n \mathrm{e}^{j\pi n}\right\} = \Delta t \text{SHIFT}\{\text{DFT}\{x_n\}\} \tag{4.43}$$

Η παραπάνω σχέση μπορεί να γραφεί και ως:

$$\text{SHIFT}\left\{\tilde{X}_k \mathrm{e}^{-j\pi k}\right\} = \text{SHIFT}\left\{\tilde{X}_k\right\} \mathrm{e}^{-j\pi k} = \Delta t \text{DFT}\{x_n\} \tag{4.44}$$

οπότε:

$$\text{SHIFT}\left\{\tilde{X}_k\right\} = \Delta t \text{DFT}\{\text{SHIFT}\{x_n\}\} \tag{4.45}$$

που τελικά δίνει

$$\tilde{X}_k = \Delta t \text{SHIFT}\{\text{DFT}\{\text{SHIFT}\{x_n\}\}\} \tag{4.46}$$

Η (4.46) μας δείχνει έναν τρόπο να υπολογίζουμε τις τιμές του μετασχηματισμού Fourier $X(f)$ σε συγκεκριμένα σημεία $X(f_k) = \tilde{X}_k$ χρησιμοποιώντας τα δείγματα του σήματος $x(t_n)$ χρησιμοποιώντας μία αλυσίδα μετασχηματισμών SHIFT - DFT - SHIFT.

Ας υλοποιήσουμε τώρα τις αντίστοιχες μεθόδους που υπολογίζουν το μετασχηματισμό Fourier του σήματος στην κλάση `signal`. Στο listing 4.11 δείχνουμε δύο μεθόδους, την `calc_spectrum` και την `calc_inv_spectrum` που υπολογίζουν τον μετασχηματισμό Fourier και τον αντίστροφο μετασχηματισμό Fourier αντίστοιχα.

```
154     def calc_spectrum(self):
155         self.Df = 1 / self.Dt / self.N
156         self.f = np.arange( -self.N / 2, self.N/2 ) * self.Df
157         self.spec = self.Dt * np.fft.fftshift(
158                 np.fft.fft(
159                         np.fft.fftshift(self.samples) ) )
160         return self.spec, self.f
161 
162     def calc_inv_spectrum(self):
163         if self.t is not None:
164             self.Dt = 1 / self.Df / self.N
165             if self.t is None:
166                 self.t = np.arange( -self.N / 2, self.N/2 ) * self.Dt
167 
168         self.samples = 1 / self.Dt * np.fft.fftshift(
169                 np.fft.ifft(
170                         np.fft.fftshift(self.spec) ) )
171         return self.samples, self.t
```

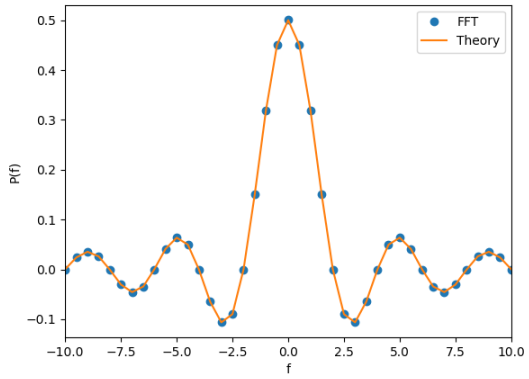
Listing 4.11: Υπολογισμός του μετασχηματισμού Fourier στην `signal`

Μπορούμε τώρα να συγκρίνουμε το αποτέλεσμα που προκύπτει από την FFT σε μία ειδική περίπτωση όπου γνωρίζουμε και θεωρητικά πως είναι ο μετασχηματισμός Fourier του σήματος. Στο παράδειγμα 4.6.1 είδαμε πως υπολογίζεται ο μετασχηματισμός Fourier του τετραγωνικού παλμού, οπότε μπορούμε να χρησιμοποιήσουμε αυτό το σήμα στις δοκιμές μας. Στο listing 4.12 κάνουμε ακριβώς αυτό και στην εικόνα 4.12 δείχνουμε το αποτέλεσμα. Βλέπουμε ότι η θεωρητική μορφή του μετασχηματισμού ταιριάζει πολύ καλά με την τιμή που παίρνουμε από τον FFT.

```
1  import commlib as cl
2  import numpy as np
3  import matplotlib.pyplot as plt
4  
5  Tmin = -1
6  Tmax = 1
7  T1 = 0.5
8  N = 1024
9  t = np.linspace(Tmin, Tmax, N)
10 
11 x = cl.square_pulse(t, T1)
12 x.calc_spectrum()
13 
14 plt.close('all')
15 
16 plt.figure()
17 plt.plot(x.f, np.real(x.spec), 'o', label = 'FFT' )
18 X_an = T1 * np.sinc(x.f * T1)
19 plt.plot(x.f, X_an, label = 'Theory')
20 plt.xlabel('f')
21 plt.ylabel('P(f)')
22 plt.legend()
23 
24 plt.xlim([-10, 10])
```

Listing 4.12: `fftcheck.py`

Έχει ενδιαφέρον να συγκρίνουμε και το αποτέλεσμα της μεθόδου σε μία δεύτερη περίπτωση όπου πάλι γνωρίζουμε τον μετασχηματισμό Fourier του σήματος και θεωρητικά.



Εικόνα 4.12: Ο μετασχηματισμός Fourier του τετραγωνικού σήματος

Παράδειγμα 4.6.2: **Μετασχηματισμός Fourier του συνημιτόνου**

Δίνεται ένα φέρον σήμα

$$x(t) = A\cos(2\pi f_0 t) \tag{4.47}$$

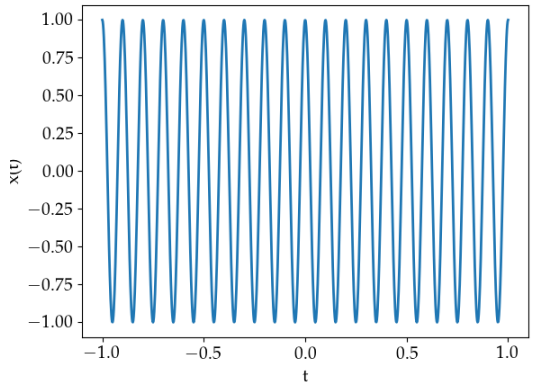
να βρείτε το μετασχηματισμό Fourier $X(f)$ θεωρητικά και να συγκρίνεται τα αποτελέσματα που λαμβάνουμε και μέσω του FFT για $f_0 = 10$.

Λύση

Ο μετασχηματισμός Fourier του σήματος στην (4.47) μπορεί να υπολογιστεί από τυπολόγια τα οποία υπάρχουν σε διάφορα βιβλία Σημάτων και Συστημάτων. Προκύπτει ότι

$$X(f) = \frac{A}{2}\left[\delta(f - f_0) + \delta(f + f_0)\right] \tag{4.48}$$

Στο listing 4.13 παραθέτουμε τον κώδικα που υπολογίζει αριθμητικά το φάσμα του φέροντος σήματος ενώ στην εικόνα 4.13 δείχνουμε το σήμα στο πεδίο του χρόνου και στην εικόνα 4.14 το σήμα στο πεδίο των συχνοτήτων. Παρατηρούμε ότι το φάσμα που υπολογίζεται αποτελείται από δύο έντονες κορυφές στις κοντά στις συχνότητες $f = -10$ και $f = +10$ όπως θα περιμέναμε και από τις συναρτήσεις $\delta(f + f_0)$ και $\delta(f - f_0)$. Επίσης το ύψος των κορυφών αυτών είναι περίπου ίσο με 1. Θυμηθείτε ότι σύμφωνα με τα όσα είπαμε στην ενότητα 3.9 η αριθμητική προσέγγιση της κάθε συνάρτησης $\frac{1}{2}\delta(f \pm f_0)$ θα πρέπει να έχει περίπου ύψος ίσο με $0.5/\Delta f$. Στην περίπτωση μας το Δf είναι το αντίστροφο του Δt που είναι σχεδόν ίσο με 2 οπότε $\Delta f \approx 0.5$ όπως φαίνεται και από τον υπολογισμό που γίνεται στο τέλος του listing 4.13. Το ύψος των κορυφών επομένως θα είναι $0.5/\Delta f \approx 1$ όπως φαίνεται δηλαδή και στην εικόνα 4.14.



Εικόνα 4.13: Το φέρον σήμα του παραδείγματος 4.6.2 στο πεδίο του χρόνου.

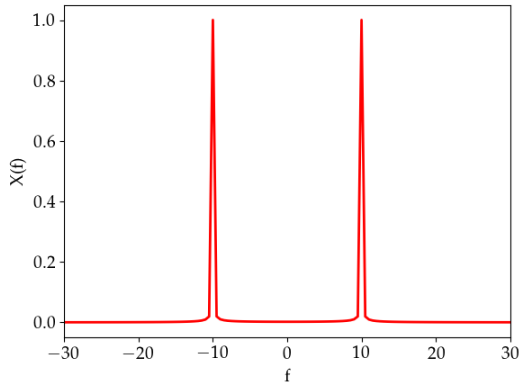
```
import commlib as cl
import numpy as np
import matplotlib.pyplot as plt

Tmin = -1
Tmax = 1
N = 1024
t = np.linspace(Tmin, Tmax, N)
f0 = 10
x = cl.carrier(t, f0)
x.calc_spectrum()

plt.close('all')
plt.figure(1)
x.plot(line_type = '-')
plt.xlabel('t')
plt.ylabel('x(t)')

plt.figure()
plt.plot(x.f, np.abs(x.spec), 'r-')
plt.xlabel('f')
plt.ylabel('X(f)')
plt.xlim([-30, 30])

print(x.f[1] - x.f[0])
```

Listing 4.13: `costest.py`

4.7 Ιδιότητες του μετασχηματισμού Fourier

Υπάρχουν μερικές χρήσιμες ιδιότητες του μετασχηματισμού Fourier που θα μας φανούν χρήσιμες στα παρακάτω. Ο μετασχηματισμός Fourier είναι γραμμικός, δηλαδή αν $x(t)$ και $y(t)$ είναι δύο σήματα με μετασχηματισμό Fourier $X(f)$ και $Y(f)$ αντίστοιχα, ο μετασχηματισμός Fourier του αθροίσματος $z(t) = x(t) + y(t)$ θα είναι το άθροισμα των αντίστοιχων μετασχηματισμών,

$$Z(f) = \mathscr{F}\{z(t)\} = \mathscr{F}\{x(t) + y(t)\} = \mathscr{F}\{x(t)\} + \mathscr{F}\{y(t)\} = X(f) + Y(f) \tag{4.49}$$



Εικόνα 4.14: Το φέρον σήμα του παραδείγματος 4.6.2 στο πεδίο των συχνοτήτων.

Επίσης όταν πολλαπλασιάζουμε ένα σήμα $x(t)$ με μία σταθερά c ο μετασχηματισμός Fourier θα είναι $cX(f)$,

$$\mathscr{F}\{cx(t)\} = c\mathscr{F}\{x(t)\} = cX(f) \tag{4.50}$$

Αν θεωρήσουμε ένα σήμα $z(t)$ που δίνεται από τον συγκερασμό δύο σημάτων $x(t)$ και $y(t)$,

$$z(t) = \int_{-\infty}^{+\infty} y(t-\tau)x(\tau)\mathrm{d}\tau \tag{4.51}$$

Το παραπάνω ολοκλήρωμα ονομάζεται *συγκερασμός* μεταξύ του $x(t)$ και του $y(t)$ και συμβολίζεται με το $*$, οπότε γράφουμε:

$$z(t) = x(t) * y(t) \tag{4.52}$$

Το φάσμα του $z(t)$ είναι το γινόμενο των επιμέρους μετασχηματισμών,

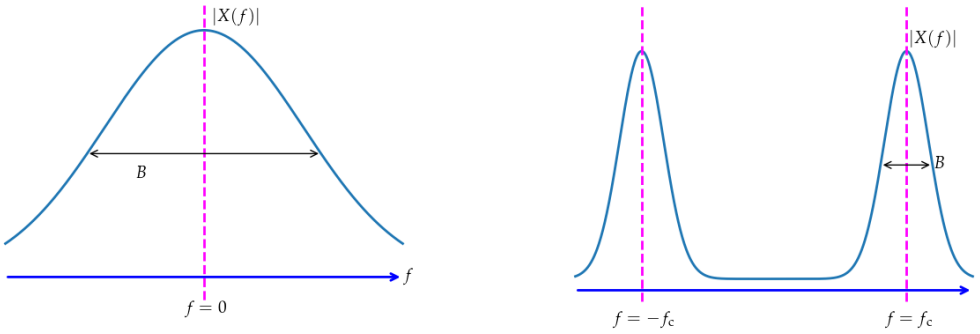
$$Z(f) = X(f)Y(f) \tag{4.53}$$

Επίσης μπορούμε να δείξουμε ότι το σήμα $y(t) = x(-t)$ έχει μετασχηματισμό Fourier το συζυγές του $X^*(f)$.

$$\mathscr{F}\{x(-t)\} = X^*(f) \tag{4.54}$$

Αν έχουμε ένα πραγματικό σήμα, τότε υπάρχει μία ενδιαφέρουσα ιδιότητα του μετασχηματισμού Fourier που θα αποδειχθεί σημαντική για την περαιτέρω ανάλυση μας. Για ένα παραγματικό σήμα $x(t)$ μπορούμε να δείξουμε ότι ισχύει $X(f) = X^*(-f)$, δηλαδή ο μετασχηματισμός Fourier στις θετικές συχνότητες $X(f)$ για $f > 0$ καθορίζει και τον μετασχηματισμό Fourier στις αρνητικές συχνότητες $X(-f)$ και μάλιστα ο ένας είναι ο μιγαδικός συζυγής του άλλου.

Αν θυμηθούμε τον ορισμό του μετασχηματισμού Fourier από την (4.24). Αν θεωρήσουμε ένα πραγματικό σήμα $x(t)$ και προσπαθήσουμε να υπολογίσουμε το μιγαδικό συζυγές $X^*(-f)$ του



(a) (b)

Εικόνα 4.15: Παραδείγματα φασμάτων σημάτων: (a) βασικής ζώνης και (b) ζωνοπερατό.

$X(-f)$ θα έχουμε:

$$X^*(-f) = \left(\int_{-\infty}^{+\infty} x(t)\mathrm{e}^{j2\pi ft}\mathrm{d}t\right)^* = \int_{-\infty}^{+\infty} \left[x(t)\mathrm{e}^{j2\pi ft}\mathrm{d}t\right]^* \mathrm{d}t = \int_{-\infty}^{+\infty} x^*(t)\mathrm{e}^{-j2\pi ft}\mathrm{d}t = \int_{-\infty}^{+\infty} x(t)\mathrm{e}^{-j2\pi ft}\mathrm{d}t = X(f) \quad (4.55)$$

Επομένως προκύπτει ότι $X^*(-f) = X(f)$. Εφόσον δύο μιγαδικοί αριθμοί που είναι συζυγείς έχουν το ίδιο μέτρο θα ισχύει

$$|X(f)| = |X(-f)| \quad (4.56)$$

Δηλαδή το μέτρο του φάσματος $|X(f)|$ θα είναι συμμετρικό γύρω από το $f = 0$. Στην εικόνα 4.15 έχουμε ζωγραφίσει δύο παραδείγματα φασμάτων που αντιστοιχούν σε πραγματικά σήματα $x(t)$. Στην πρώτη περίπτωση στην εικόνα 4.15a, το φάσμα είναι συγκεντρωμένο στις χαμηλές συχνότητες γύρω από το $f = 0$. Στην περίπτωση λέμε πως έχουμε ένα σήμα *βασικής ζώνης*. Στην δεύτερη περίπτωση, στην εικόνα 4.15b, το φάσμα $X(f)$ είναι συγκεντρωμένο γύρω από μία υψηλή συχνότητα $f = f_c$ οπότε έχουμε ένα *ζωνοπερατό* σήμα.

Σήματα βασικής ζώνης συναντάει κανείς σε ενσύρματα συστήματα όπως για παράδειγμα το xDSL (Digital Subscriber Line - Ψηφιακή Γραμμή Συνδρομητή). Στα ασύρματα συστήματα, έχουμε συνήθως ζωνοπερατάτα σήματα. Στην περίπτωση για παράδειγμα των συστημάτων WiFi η κεντρική συχνότητα είναι $f_c = 2.4$ GHz ή $f_c = 5$ GHz. Για τα ζωνοπερατά σήματα, συνήθως το φάσμα είναι πολύ στενό γύρω από την κεντρική συχνότητα $f = f_c$ οπότε αν B το εύρος του φάσματος (περισσότερα για αυτό και στην ενότητα 4.9) θα έχουμε:

$$\frac{B}{f_c} << 1 \quad (4.57)$$

4.8 Ενέργεια στο πεδίο των συχνοτήτων

Είναι ενδιαφέρον να παρατηρήσουμε ότι μπορούμε να υπολογίζουμε την ενέργεια τόσο στο πεδίο του χρόνου όσο και στο πεδίο των συχνοτήτων. Υπάρχει μία ταυτότητα που ονομάζεται

ταυτότητα του Parseval σύμφωνα με την οποία:

$$\int_{-\infty}^{\infty} |x(t)|^2 \mathrm{d}t = \int_{-\infty}^{\infty} |X(f)|^2 \mathrm{d}f \tag{4.58}$$

όπου το $X(f)$ είναι ο μετασχηματισμός Fourier του $x(t)$. Μπορούμε να δείξουμε αριθμητικά την (4.58) χρησιμοποιώντας το listing 4.14 όπου υπολογίζουμε τα δύο μέλη της (4.58) στην περίπτωση ενός τετραγωνικού παλμού με διάρκεια $T_1 = 1$. Από τα αποτελέσματα προκύπτει ότι $\int_{-\infty}^{\infty} |x(t)|^2 \mathrm{d}t \approx 1$ και $\int_{-\infty}^{\infty} |X(f)|^2 \mathrm{d}f \approx 1$ οπότε ισχύει η (4.58).

Για να αποδείξουμε και θεωρητικά την (4.58) χρησιμοποιούμε τον ορισμό του μετασχηματισμού Fourier (4.24) στο $\int_{-\infty}^{\infty} |X(f)|^2 \mathrm{d}f$,

$$\int_{-\infty}^{\infty} |X(f)|^2 \mathrm{d}f = \int_{-\infty}^{\infty} \left| \int_{-\infty}^{+\infty} x(t) \mathrm{e}^{-j2\pi ft} \mathrm{d}t \right|^2 \mathrm{d}f \tag{4.59}$$

Για το εσωτερικό ολοκλήρωμα θα έχουμε:

$$\left| \int_{-\infty}^{+\infty} x(t) \mathrm{e}^{-j2\pi ft} \mathrm{d}t \right|^2 = \left(\int_{-\infty}^{+\infty} x(t_1) \mathrm{e}^{-j2\pi ft_1} \mathrm{d}t_1 \right) \left(\int_{-\infty}^{+\infty} x(t_2) \mathrm{e}^{-j2\pi ft_2} \mathrm{d}t_2 \right)^* = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x(t_1) x^*(t_2) \mathrm{e}^{-j2\pi (t_1 - t_2) f} \mathrm{d}t_1 \mathrm{d}t_2 \tag{4.60}$$

Συνδυάζοντας τις (4.59) και (4.60) θα έχουμε:

$$\int_{-\infty}^{\infty} |X(f)|^2 \mathrm{d}f = \int_{-\infty}^{\infty} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x(t_1) x^*(t_2) \mathrm{e}^{-j2\pi (t_1 - t_2) f} \mathrm{d}t_1 \mathrm{d}t_2 \mathrm{d}f = \int_{-\infty}^{\infty} \int_{-\infty}^{+\infty} x(t_1) x^*(t_2) \left(\int_{-\infty}^{+\infty} \mathrm{e}^{-j2\pi (t_1 - t_2) f} \mathrm{d}f \right) \mathrm{d}t_1 \mathrm{d}t_2 \tag{4.61}$$

Όσο και να φαίνεται απίστευτο, το παραπάνω ολοκλήρωμα μπορεί να υπολογιστεί σχετικά εύκολα. Ξεκινάμε με το εσωτερικό ολοκλήρωμα

```python
from commlib import square_pulse
import numpy as np

T = 10
T1 = 1

t = np.linspace(-T/2, T/2, 1024)
x = square_pulse(t, T1)
Et = x.energy()
x.calc_spectrum()
Ef = np.trapz(np.abs(x.spec) ** 2.0, x.f)

print('Energy in time domain', Et)
print('Energy in frequency domain', Ef)
```

Listing 4.14: `parseval.py`

$$\int_{-\infty}^{+\infty} \mathrm{e}^{-j2\pi (t_1 - t_2) f} \mathrm{d}f = \int_{-\infty}^{+\infty} 1 \times \mathrm{e}^{j2\pi (t_2 - t_1) f} \mathrm{d}f \tag{4.62}$$

Η (4.62) είναι στην ουσία ο αντίστροφος μετασχηματισμός της μονάδας, 1, υπολογισμένος στο $t = t_2 - t_1$. Αν υπολογίσουμε τον μετασχηματισμό Fourier $\Delta(f)$ της συνάρτησης $\delta(t)$ μπορούμε να δείξουμε ότι:

$$\Delta(f) = \int_{-\infty}^{+\infty} \delta(t) \mathrm{e}^{-j2\pi ft} \mathrm{d}t = \mathrm{e}^{-j2\pi 0t} = 1 \tag{4.63}$$

όπου έχουμε χρησιμοποιήσει την βασική ιδιότητα της συνάρτησης $\delta(t)$ που είδαμε στην (3.41) και επαναλαμβάνουμε εδώ:

$$\int_{-\infty}^{+\infty} x(t)\delta(t)\mathrm{d}t = x(0) \tag{4.64}$$

Οπότε ο μετασχηματισμός Fourier της $\delta(t)$ είναι το 1 και επομένως ο αντίστροφος μετασχηματισμός του 1 θα είναι το $\delta(t)$,

$$\int_{-\infty}^{+\infty} \Delta(f) \mathrm{e}^{j2\pi ft} \mathrm{d}f = \int_{-\infty}^{+\infty} \mathrm{e}^{j2\pi ft} \mathrm{d}f = \delta(t) \tag{4.65}$$

Αν θέσουμε $t = t_2 - t_1$, θα έχουμε:

$$\int_{-\infty}^{+\infty} \mathrm{e}^{j2\pi f(t_2-t_1)} \mathrm{d}f = \delta(t_2 - t_1) \tag{4.66}$$

Αντικαθιστώντας στην (4.60),

$$\begin{aligned} \int_{-\infty}^{\infty} |X(f)|^2 \mathrm{d}f &= \int_{-\infty}^{\infty} \int_{-\infty}^{+\infty} x(t_1)x^*(t_2)\delta(t_2 - t_1)\mathrm{d}t_1\mathrm{d}t_2 = \\ &= \int_{-\infty}^{\infty} \left(\int_{-\infty}^{+\infty} x(t_1)x^*(t_2)\delta(t_2 - t_1)\mathrm{d}t_1 \right) \mathrm{d}t_2 = \int_{-\infty}^{+\infty} x(t_2)x^*(t_2)\mathrm{d}t_2 = \int_{-\infty}^{+\infty} |x(t_2)|^2 \mathrm{d}t_2 \end{aligned} \tag{4.67}$$

Με την (4.67), αποδείξαμε την ταυτότητα του Parseval. Χρησιμοποιώντας την ταυτότητα του Parseval μπορούμε να ορίσουμε ένα πολύ χρήσιμο μέγεθος που ονομάζεται φασματική πυκνότητα ισχύος (power spectral density - PSD). Η μέση ισχύς δίνεται από την σχέση (3.25), την οποία επαναλαμβάνουμε και εδώ:

$$\bar{P} = \lim_{\tau \to \infty} \frac{1}{\tau} \int_{-\tau/2}^{\tau/2} |x(t)|^2 \mathrm{d}t \tag{4.68}$$

Αν ορίσουμε το σήμα $x_\tau(t)(t)$ ως:

$$x_\tau(t) = \begin{cases} x(t) & , |t| \leq \frac{\tau}{2} \\ 0 & , \text{διαφορετικά} \end{cases} \tag{4.69}$$

Το σήμα $x_\tau(t)$ στην (4.69) είναι μία "παραθυρωμένη" έκδοση του $x(t)$ στο διάστημα $[-\tau/2, \tau/2]$. Χρησιμοποιώντας το $x_\tau(t)$ μπορούμε να γράψουμε:

$$\frac{1}{\tau} \int_{-\tau/2}^{\tau/2} x^2(t)\mathrm{d}t = \frac{1}{\tau} \int_{-\tau/2}^{\tau/2} x_\tau^2(t)\mathrm{d}t = \frac{1}{\tau} \int_{-\infty}^{+\infty} x_\tau^2(t)\mathrm{d}t = \frac{1}{\tau} \int_{-\infty}^{+\infty} |X_\tau(f)|^2 \mathrm{d}f \tag{4.70}$$

οπότε η μέση ισχύς στην (4.68) γράφεται:

$$\bar{P} = \int_{-\infty}^{+\infty} \lim_{\tau\to\infty} \frac{1}{\tau}|X_\tau(f)|^2 \mathrm{d}f \tag{4.71}$$

Το όρισμα του ολοκληρώματος

$$S_x(f) = \lim_{\tau\to\infty} \frac{1}{\tau}|X_\tau(f)|^2 \tag{4.72}$$

ονομάζεται *φασματική πυκνότητα ισχύος* (power spectral density - PSD) του $x(t)$. Η μέση ισχύς του σήματος συνδέεται με την PSD σύμφωνα με την σχέση:

$$\bar{P} = \int_{-\infty}^{+\infty} S_X(f)\mathrm{d}f \tag{4.73}$$

Η (4.73) μας υποδεικνύει ότι η $S_X(f)$ χαρακτηρίζει την σημασία που έχουν στο υπολογισμό της μέσης ισχύος $\bar{P}$ οι διάφορες φασματικές συνιστώσες του σήματος. Ας προσπαθήσουμε να υπολογίσουμε τη φασματική πυκνότητα ισχύος στην περίπτωση του σήματος (4.47). Το σήμα αυτό είναι περιοδικό με περίοδο $T = 1/f_0$, οπότε ισχύει $x(t+T) = x(t)$. Ο μετασχηματισμός Fourier του $x_\tau(t)$ υπολογίζεται ως εξής:

$$X_\tau(f) = \int_{-\infty}^{+\infty} x_\tau(t)\mathrm{e}^{-j2\pi ft}\mathrm{d}t = \int_{-\tau/2}^{+\tau/2} x(t)\mathrm{e}^{-j2\pi ft}\mathrm{d}t = \int_{-\tau/2}^{+\tau/2} A\cos(2\pi f_0 t)\mathrm{e}^{-j2\pi ft}\mathrm{d}t \tag{4.74}$$

Χρησιμοποιούμε στη συνέχεια την ταυτότητα:

$$\cos\phi = \frac{1}{2}\left(\mathrm{e}^{j\phi} + \mathrm{e}^{-j\phi}\right) \tag{4.75}$$

και το φάσμα $X_\tau(t)$ γράφεται:

$$X_\tau(f) = \int_{-\tau/2}^{+\tau/2} A\cos(2\pi f_0 t)\mathrm{e}^{-j2\pi ft}\mathrm{d}t = \frac{A}{2}\int_{-\tau/2}^{+\tau/2}\left(\mathrm{e}^{-j2\pi(f+f_0)t} + \mathrm{e}^{-j2\pi(f-f_0)t}\right)\mathrm{d}t \tag{4.76}$$

Για τον υπολογισμό των ολοκληρωμάτων χρησιμοποιούμε τον γνωστό τύπο:

$$\int \mathrm{e}^{j\alpha t}\mathrm{d}t = \frac{\mathrm{e}^{j\alpha t}}{j\alpha} + C \tag{4.77}$$

οπότε

$$X_\tau(f) = \frac{A}{2}\frac{\mathrm{e}^{-j\pi(f+f_0)\tau} - \mathrm{e}^{j\pi(f+f_0)\tau}}{-j\pi(f+f_0)} + \frac{A}{2}\frac{\mathrm{e}^{-j\pi(f-f_0)\tau} - \mathrm{e}^{j\pi(f-f_0)\tau}}{-j\pi(f-f_0)} \tag{4.78}$$

Μπορούμε να χρησιμοποιήσουμε την σχέση

$$\sin\phi = \frac{1}{2j}\left(\mathrm{e}^{j\phi} - \mathrm{e}^{-j\phi}\right) \tag{4.79}$$

για να απλοποιήσουμε λίγο την (4.78),

$$X_\tau(f) = \frac{A}{2}\frac{\sin(\pi(f+f_0)\tau)}{\pi(f+f_0)} + \frac{A}{2}\frac{\sin(\pi(f-f_0)\tau)}{\pi(f-f_0)} \tag{4.80}$$

Η συνάρτηση sinc ορίζεται ως $\sin(\pi x)/(\pi x)$, οπότε:

$$X_\tau(f) = \frac{A\tau}{2}\text{sinc}((f+f_0)\tau) + \frac{A\tau}{2}\text{sinc}((f-f_0)\tau) \tag{4.81}$$

Στο listing 4.15 δείχνουμε τον κώδικα για να κάνουμε την γραφική παράσταση του $X_\tau(f)$ που υπολογίσαμε στην (4.81) για διάφορες τιμές του τ. Στην εικόνα 4.16 δείχνουμε το $X_\tau(f)$. Παρατηρούμε ότι όσο μεγαλώνει το τ τόσο πιο ψηλές και στενές γίνονται οι κορυφές της $X_\tau(f)$ στο $f = \pm f_0$ κάτι που υποδεικνύει ότι στο όριο $\tau \to \infty$, θα έχουμε μία συμπεριφορά παρόμοια με τις συναρτήσεις $\delta(f \pm f_0)$. Πράγματι στην βιβλιογραφία μπορεί κανείς να βρει την εξής ιδιότητα των συναρτήσεων δ,

$$\lim_{\alpha\to\infty} \alpha\text{sinc}(\alpha x) = \delta(x) \tag{4.82}$$

```python
import numpy as np
import matplotlib.pyplot as plt

T = 10
f0 = 1
Nf = 10000

taus = np.array([1, 2, 10])
f = np.linspace(-2*f0, 2*f0, Nf)

plt.close('all')
for tau in taus:
    Xtau = tau * np.sinc( 2 * (f+f0) * tau ) + tau * np.sinc( 2 * (f-f0) * tau )
    plt.figure(1)
    plt.plot(f, Xtau, label = str(tau))

    SXtau = np.abs(Xtau) ** 2.0 / tau
    plt.figure(2)
    plt.plot(f, SXtau, label = str(tau))

plt.figure(1)
plt.legend()
plt.xlabel('f')
plt.ylabel('X(f)')

plt.figure(2)
plt.legend()
plt.xlabel('f')
plt.ylabel('SX(f)')
```

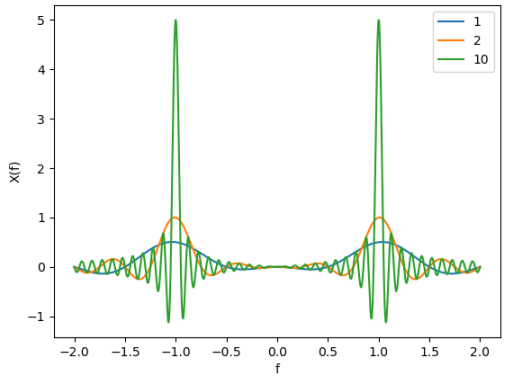
Listing 4.15: `psd.py`

οπότε όταν $\tau \to \infty$, θα έχουμε:

$$\lim_{\tau\to\infty} X_\tau(f) = \frac{A}{2}\delta(f-f_0) + \frac{A}{2}\delta(f+f_0) \tag{4.83}$$

Αν δείτε το αποτέλεσμα της (4.83), συμπίπτει με αυτό του 4.6.2. Επίσης θα έχουμε:

$$\frac{|X_\tau(f)|^2}{\tau} = \frac{A^2\tau}{4}\left(\text{sinc}((f+f_0)\tau) + \text{sinc}((f-f_0)\tau)\right)^2 =$$
$$\frac{A^2\tau}{4}\left(\text{sinc}^2((f+f_0)\tau) + 2\text{sinc}((f-f_0)\tau)\text{sinc}((f+f_0)\tau) + \text{sinc}^2((f-f_0)\tau)\right) \tag{4.84}$$



Εικόνα 4.16: Η γραφική παράσταση της $X_\tau(f)$ για διάφορες τιμές του τ.

Για να υπολογίσουμε την PSD θα πρέπει να χρησιμοποιήσουμε την (4.72) οπότε και όρια της μορφής $\alpha \text{sinc}^2(\alpha x)$ όταν $\alpha \to \infty$. Ας ορίσουμε την συνάρτηση:

$$f_\alpha(x) = \alpha \text{sinc}^2(\alpha x) \tag{4.85}$$

και ας την μελετήσουμε με το listing 4.16 όπου κάνουμε καταρχήν την γραφική της παράσταση της $f_\alpha(x)$ για α ίσο με 1, 4 και 10 η οποία φαίνεται στην εικόνα 4.17. Παρατηρούμε ότι όσο αυξάνει το α τόσο ψηλώνει αλλά και στενεύει η κορυφή στο $x = 0$. Στη συνέχεια για περισσότερες τιμές της παραμέτρου α υπολογίζουμε και το εμβαδό της συνάρτησης,

$$E_\alpha = \int_{-\infty}^{\infty} f_\alpha(x) \text{d}x \tag{4.86}$$

χρησιμοποιώντας τον κανόνα του τραπεζίου. Το αποτέλεσμα του αριθμητικού υπολογισμού για το E_α το βλέπουμε στην εικόνα 4.18. Παρατηρούμε ότι $E_\alpha \approx 1$ και η προσέγγιση γίνεται καλύτερη όσο μεγαλώνει το α. Σύμφωνα με τα παραπάνω, η $f_\alpha(x)$ είναι μία συνάρτηση που όσο αυξάνει το α γίνεται όλο και πιο στενή αλλά το εμβαδόν της είναι ίσο με την μονάδα όπως και στην περίπτωση της προσέγγισης της συνάρτησης $\delta(x)$ που είδαμε στην εικόνα 3.15. Αριθμητικά επομένως προκύπτει ότι:

$$\lim_{\alpha \to \infty} f_\alpha(x) = \delta(x) \tag{4.87}$$

```
import numpy as np
import matplotlib.pyplot as plt

alphas = np.array([1, 4, 10])

Nx = 100000
x = np.linspace(-3,+3, Nx)
Na = 100

plt.close('all')
for alpha in alphas:
```

```
    falpha = alpha * np.sinc( alpha * x ) ** 2.0
    plt.figure(1)
    plt.plot(x, falpha, label = str(alpha) )


plt.legend()
plt.xlabel('x')
plt.ylabel('fa(x)')

alphas = np.linspace(1, 100, Na)
Ea = np.zeros(alphas.size)

for i, alpha in enumerate(alphas):
    falpha = alpha * np.sinc( alpha * x ) ** 2.0
    E = np.trapz(falpha, x)
    Ea[i] = E

plt.figure(2)
plt.plot(alphas, Ea)
plt.xlabel('a')
plt.ylabel('Ea')
```

Listing 4.16: `sinca2.py`

Λαμβάνοντας το όριο της (4.84) όταν $\tau \to \infty$ προκύπτει ένα όριο της μορφής $\tau \text{sinc}^2((f+f_0)\tau)$ από τον πρώτο όρο, ένα όριο της μορφής $\tau \text{sinc}^2((f-f_0)\tau)$ από τον τρίτο όρο και από τον δεύτερο όρο ένα όριο της μορφής:

$$\lim_{\tau\to\infty} \{\tau \text{sinc}((f-f_0)\tau)\text{sinc}((f+f_0)\tau)\} \tag{4.88}$$

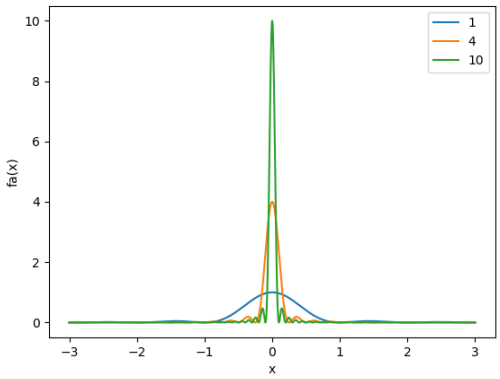
Στο listing 4.16 κάνουμε την γραφική παράσταση της συνάρτησης $F_\tau(f) = \tau \text{sinc}((f-f_0)\tau)\text{sinc}((f+f_0)\tau)$ για $f_0 = 1.5$ που φαίνεται στην εικόνα 4.19. Παρατηρούμε ότι δεν έχουμε μία συμπεριφορά όπως είχαμε στην εικόνα 4.17, δηλαδή όσο μεγαλώνει το τ, οι κορυφές της $F_\tau(f)$ στενεύουν αλλά δεν ανεβαίνουν σε τιμή. Όταν $\tau \to \infty$ οι κορυφές της συνάρτησης θα γίνονται πολύ στενές με αποτέλεσμα να μπορούμε να αγνοήσουμε την συνάρτηση και να θεωρήσουμε ότι:

$$\begin{aligned} S_X(f) &= \lim_{\tau\to\infty} \frac{|X_\tau(f)|^2}{\tau} = \\ &\lim_{\tau\to\infty} \left\{ \frac{A^2\tau}{4} \left(\text{sinc}^2((f+f_0)\tau) + 2\text{sinc}((f-f_0)\tau)\text{sinc}((f+f_0)\tau) + \text{sinc}^2((f-f_0)\tau) \right) \right\} \approx \\ &\lim_{\tau\to\infty} \left\{ \frac{A^2\tau}{4} \left(\text{sinc}^2((f+f_0)\tau) + \text{sinc}^2((f-f_0)\tau) \right) \right\} = \frac{A^2}{4} \left(\delta(f-f_0) + \delta(f+f_0) \right) \end{aligned} \tag{4.89}$$

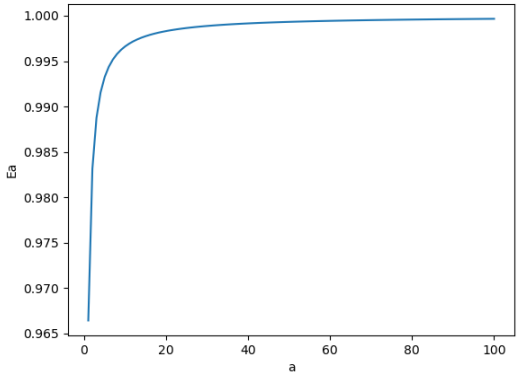
Σύμφωνα με την (4.89), η PSD του σήματος δίνεται από το άθροισμα δύο συναρτήσεων $\delta(f \pm f_0)$ με πλάτος $\frac{A^2}{4}$. Η μέση ισχύς προκύπτει από το ολοκλήρωμα της $S_X(f)$, δηλαδή:

$$\bar{P} = \int_{-\infty}^{\infty} S_X(f) \mathrm{d}f = \frac{A^2}{4} + \frac{A^2}{4} = \frac{A^2}{2} \tag{4.90}$$

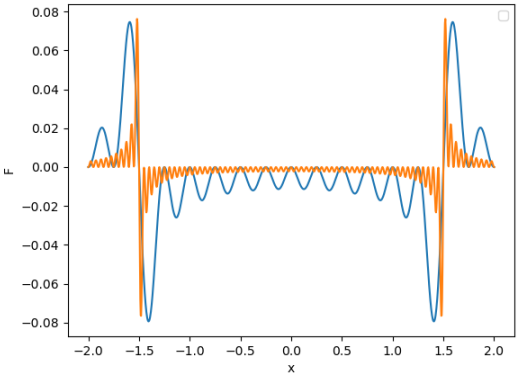
Το αποτέλεσμα αυτό είναι σύμφωνο με την (3.31).



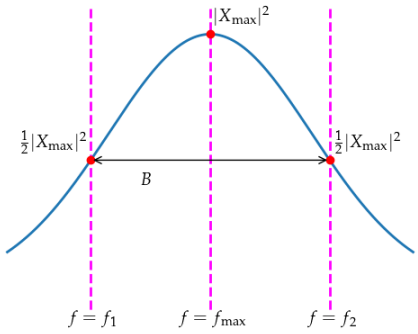
Εικόνα 4.17: Η γραφική παράσταση της $f_\alpha(x)$ για διάφορες τιμές του α.



Εικόνα 4.18: Η γραφική παράσταση του E_α για διάφορες τιμές του α.



Εικόνα 4.19: Η γραφική παράσταση της $F_\tau(x) = \tau \text{sinc}((f - f_0)\tau)\text{sinc}((f + f_0)\tau)$ για διάφορες τιμές του τ.



Εικόνα 4.20: Υπολογισμός του εύρους ζώνης του σήματος.

4.9 Εύρος Ζώνης

Ο μετασχηματισμός Fourier $X(f)$ ενός σήματος $x(t)$ καθορίζει το *φάσμα* του σήματος, δηλαδή ποιες συχνότητες f παίζουν σημαντικό ρόλο όταν το αναλύουμε στο πεδίο των συχνοτήτων. Με τον όρο *εύρος ζώνης*, αναφερόμαστε σε κάποιο συχνοτικό εύρος B μέσα στο οποίο θεωρούμε ότι περιέχεται η πλειοψηφία της ενέργειας του σήματος. Εναλλακτικά μπορούμε να ορίσουμε το πλήρες φασματικό εύρος στα L dB. Στην εικόνα 4.20 βλέπουμε ένα παράδειγμα γραφικής παράστασης του $|X(f)|^2$ που έχει μέγιστο στην συχνότητα $f = f_{max}$, ενώ στις συχνότητες $f = f_1$ και $f = f_2$, το $|X(f)|^2$ έχει μειωθεί κατά 50% σε σχέση με το $|X(f_{max})|^2$ (δηλαδή κατά -3dB). Οπότε το πλήρες φασματικό εύρος στα $L = 3$dB είναι $B_{3dB} = f_2 - f_1$. Η κατάσταση είναι παρόμοια με αυτή της εικόνας 3.13 μόνο που αντί για το πεδίο του χρόνου τώρα θεωρούμε το πεδίο των συχνοτήτων.

```
175     def bandwidth_at(self, L = 3):
176         if self.spec is None:
177             self.calc_spectrum()
178 
179         return level(self.f, np.abs(self.spec) ** 2.0, lvl = L)
```

Listing 4.17: `signal.bandwitdh_at`

```
1  from commlib import signal
2  import numpy as np
3  import matplotlib.pyplot as plt
4  
5  plt.rcParams.update({
6      "text.usetex": True,
7      "font.family": "serif",
8      "font.serif": ["Palatino"],
9      "font.size" : 14,
10     "lines.linewidth" : 2,
11 })
12 
13 
14 N = 16384
15 Tgs = np.linspace(0.1, 1, 100)
```

```
Bs = np.zeros( Tgs.size )
Ts = np.zeros( Tgs.size )

L = 3

for i, Tg in enumerate(Tgs):
    t = np.linspace(-10 * Tg, 10 * Tg, N)
    samples = np.exp( -t**2.0 / 2 / Tg ** 2.0)
    p = signal(t = t, samples = samples)
    Ts[i] = p.fwm_at( L )
    Bs[i] = p.bandwidth_at( L )

plt.close('all')
plt.figure(1)
plt.plot(Tgs, Bs, label = r'$B_\mathrm{3dB}$')
plt.plot(Tgs, Ts, label = r'$T_\mathrm{FWHM}$')
plt.ylabel('Width')
plt.xlabel('$T_\mathrm{g}$')
plt.legend()
```

Listing 4.18: `bandwidthgauss.py`

Εφαρμόζουμε την συνάρτηση `level` της `commlib` για να υλοποιήσουμε την μέθοδο `bandwidth_at` της κλάσης `signal` η οποία υπολογίζει το εύρος ζώνης του σήματος. Το listing 4.17 δείχνει πως υλοποιείται η `bandwidth_at` ενώ το listing 4.18 δείχνει ένα παράδειγμα χρήσης της. Θεωρούμε έναν Gaussian παλμό:

$$x(t) = \exp\left(-\frac{t^2}{2T_g^2}\right) \tag{4.91}$$

Στην εικόνα 4.21 δείχνουμε το πλήρες εύρος μισής ισχύος T_{FWHM} και το εύρος ζώνης B_{3dB}. Βλέπουμε ότι το T_{FWHM} μεταβάλλεται γραμμικά με το T_g ενώ το εύρος ζώνης B_{3dB} μειώνεται όσο αυξάνει το T_g (και επομένως και το T_{FWHM}). Η εξάρτηση του T_{FWHM} με το T_g δικαιολογείται από την σχέση (3.40) που είδαμε στην παράγραφο 3.8,

$$T_{\text{FWHM}} = 2T_g\sqrt{\ln 2} \tag{4.92}$$

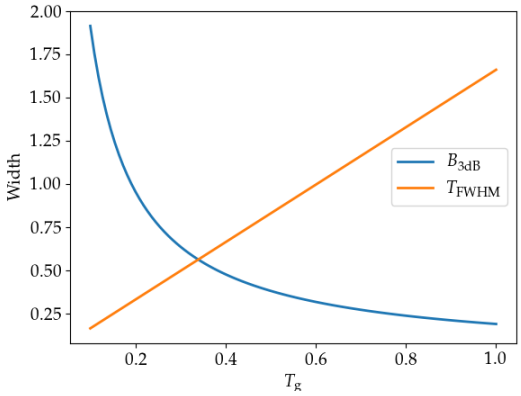
Η εξάρτηση του B_{3dB} δικαιολογείται αν θεωρήσουμε το φάσμα $X(f)$ του $x(t)$. Αν ανατρέξουμε σε πίνακες μετασχηματισμών και να δούμε ότι ο μετασχηματισμός Fourier δίνεται από την σχέση:

$$X(f) = T_g\sqrt{2}\exp\left(-2\pi^2 f^2 T_g^2\right) \tag{4.93}$$

Αν θέσουμε $B_g = 1/(2\pi T_g)$ τότε η (4.93) γράφεται ως εξής:

$$X(f) = \frac{1}{B_g\sqrt{2\pi}}\exp\left(-\frac{f^2}{2B_g^2}\right) \tag{4.94}$$

Παρατηρούμε ότι το φάσμα (4.94) έχει και αυτή μία Gaussian εξάρτηση όπως και η 4.91 με την διαφορά ότι έχει προστεθεί ένας σταθερός όρος $1/(B_g\sqrt{2\pi})$ πριν το εκθετικό και πως το T_g



Εικόνα 4.21: Η σχέση μεταξύ T_{FWHM} και $B_{3\text{dB}}$.

έχει αντικατασταθεί από το B_g. Επομένως επαναλαμβάνοντας την διαδικασία της παραγράφου 3.8 για να δείξουμε ότι το $B_{3\text{dB}}$ δίνεται από μία σχέση που είναι παρόμοια με την (4.92)

$$B_{3\text{dB}} = 2B_g\sqrt{\ln 2} = \frac{\sqrt{\ln 2}}{\pi T_g} = \frac{2\ln 2}{\pi T_{\text{FWHM}}} \tag{4.95}$$

Από την (4.95), προκύπτει ότι το $B_{3\text{dB}}$ είναι αντιστρόφως ανάλογο με το T_{FWHM}, δικαιολογώντας και την συμπεριφορά της εικόνας 4.21. Επομένως όσο πιο στενός είναι ο παλμός $x(t)$ τόσο πιο ευρύ είναι το φάσμα $X(f)$. Αυτό είναι μία γενικότερη διαπίστωση ανεξάρτητα από την μορφή του παλμού: όσο πιο μικρή είναι η διάρκεια του παλμού τόσο πιο μεγάλο εύρος ζώνης χρειαζόμαστε.

Θυμηθείτε ότι ένα ψηφιακό σήμα μπορεί να γραφεί σύμφωνα με την (3.15) ως εξής,

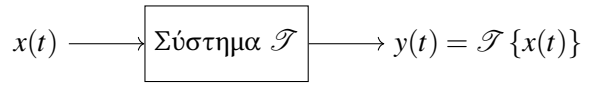
$$x(t) = \sum_{k=0}^{N} x_k p(t - kT_S) \tag{4.96}$$

όπου T_S είναι η διάρκεια του συμβόλου και $p(t)$ ένας τετραγωνικός παλμός διάρκειας T_S. Όταν το ψηφιακό σήμα είναι υψηλού ρυθμού, δηλαδή τα σύμβολα του εναλλάσσονται με γρήγορο ρυθμό, οπότε το T_S είναι μικρό, έπεται ότι οι παλμοί $p(t)$ θα έχουν μικρή διάρκεια και επομένως το σήμα θα έχει μεγάλο εύρος ζώνης. Τα παραπάνω αντανακλούν το ότι σε ένα σύστημα επικοινωνιών, η αύξηση του ρυθμού μετάδοσης έχει ως άμεσο αντίκτυπο την διεύρυνση του εύρους ζώνης που απαιτείται.

4.10 Συστήματα

Πολλές φορές στις επικοινωνίες, θεωρούμε ότι τα σήματα μας περνάνε από διάφορα συστήματα όπως καλώδια, ενισχυτές κτλ. Στην εικόνα 4.22 δείχνουμε μία βασική αναπαράσταση ενός συστήματος με μία είσοδο, το σήμα $x(t)$ και μία έξοδο το σήμα $y(t)$. Τα συστήματα αλλάζουν το σήμα εισόδου $x(t)$ και το μετασχηματίζουν στο $y(t)$. Αυτή η πράξη του μετασχηματισμού συμβολίζεται με το $\mathscr{T}$ οπότε γράφουμε

$$y(t) = \mathscr{T}\{x(t)\} \tag{4.97}$$



Εικόνα 4.22: Η γενική αναπαράσταση ενός συστήματος

Ο τρόπος με τον οποίο επιδρούν τα συστήματα στα σήματα μπορεί να είναι πολύ απλός ή και πολύ πολύπλοκος ανάλογα με τη φύση του συστήματος και το επίπεδο ακρίβειας που επιθυμούμε. Μία χρήσιμη κατηγορία σημάτων είναι τα *γραμμικά αναλλοίωτα* (linear time invariant - LTI) συστήματα. Ένα σύστημα λέγεται *γραμμικό* όταν διατηρεί την γραμμικότητα της εισόδου του στην έξοδο. Έτσι αν έχουμε δύο σήματα $x_1(t)$ και $x_2(t)$ που έχουν εξόδους $y_1(t)$ και $y_2(t)$:

$$y_1(t) = \mathscr{T}\{x_1(t)\} \tag{4.98}$$

$$y_2(t) = \mathscr{T}\{x_2(t)\} \tag{4.99}$$

τότε το άθροισμα των εισόδων τους $x_1(t)+x_2(t)$ θα αντιστοιχεί στο άθροισμα των εξόδων τους (όταν είναι στην είσοδο μόνα τους στο σύστημα), δηλαδή $y_1(t)+y_2(t)$. Δηλαδή

$$\mathscr{T}\{x_1(t)+x_2(t)\} = \mathscr{T}\{x_1(t)\} + \mathscr{T}\{x_2(t)\} = y_1(t)+y_2(t) \tag{4.100}$$

Η (4.100) μπορεί να φαίνεται προφανής αλλά δεν ισχύει για όλα τα συστήματα. Πάρτε για παράδειγμα το σύστημα:

$$\mathscr{T}\{x(t)\} = x^2(t) \tag{4.101}$$

Το παραπάνω σύστημα τετραγωνίζει το σήμα στην είσοδο και το στέλνει στην έξοδο. Μπορούμε να δούμε ότι:

$$\mathscr{T}\{x_1(t)+x_2(t)\} = (x_1(t)+x_2(t))^2 = x_1^2(t)+x_2^2(t)+2x_1(t)x_2(t) \tag{4.102}$$

ενώ έχουμε:

$$\mathscr{T}\{x_1(t)\} + \mathscr{T}\{x_2(t)\} = x_1^2(t)+x_2^2(t) \tag{4.103}$$

Συγκρίνοντας τις δύο παραπάνω σχέσεις καταλαβαίνουμε ότι μόνο εάν $x_1(t) = 0$ ή $x_2(t) = 0$ θα ισχύει:

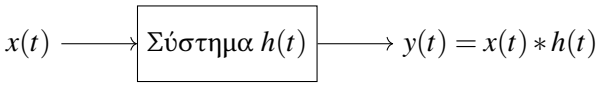
$$\mathscr{T}\{x_1(t)+x_2(t)\} = \mathscr{T}\{x_1(t)\} + \mathscr{T}\{x_2(t)\} \tag{4.104}$$

οπότε το σύστημα μας δεν είναι γραμμικό. Υπάρχει και μία συμπληρωματική ιδιότητα που πρέπει να ισχύει για να είναι γραμμικό το σύστημα μας. Για κάθε σταθερά c θα πρέπει να ισχύει:

$$\mathscr{T}\{cx(t)\} = c\mathscr{T}\{x(t)\} \tag{4.105}$$

Είναι εύκολο να δούμε ότι ούτε αυτή η ιδιότητα ισχύει για το προηγούμενο σύστημα που εξετάσαμε.

Μία άλλη κατηγορία συστημάτων που μας ενδιαφέρουν στις επικοινωνίες είναι τα *αναλλοίωτα συστήματα.* Πρόκειται για συστήματα των οποίων η συμπεριφορά δεν αλλάζει με



Εικόνα 4.23: Αναπαράσταση ενός LTI συστήματος

τον χρόνο. Έτσι αν εφαρμόσουμε το ίδιο σήμα εισόδου $x(t)$ μετατοπισμένο κατά t_0 η απόκριση του συστήματος $y(t) = \mathscr{T}\{x(t)\}$ απλά θα είναι μετατοπισμένη και αυτή κατά t_0. Οπότε θα έχουμε:

$$\mathscr{T}\{x(t-t_0)\} = y(t-t_0) \tag{4.106}$$

Το προηγούμενο σύστημα $\mathscr{T}\{x(t)\} = x^2(t)$ είναι χρονικά αναλλοίωτο εφόσον

$$\mathscr{T}\{x(t-t_0)\} = x^2(t-t_0) = y(t-t_0) \tag{4.107}$$

Αντίθετα αν είχαμε ένα σύστημα που μετασχηματίζει το σήμα ως εξής:

$$\mathscr{T}\{x(t)\} = \mathrm{e}^{-t}x^2(t) \tag{4.108}$$

τότε θα είχαμε:

$$\mathscr{T}\{x(t-t_0)\} = \mathrm{e}^{-t}x^2(t-t_0) = \mathrm{e}^{-t_0}\mathrm{e}^{-(t-t_0)}x^2(t-t_0) = \mathrm{e}^{-t_0}y(t-t_0) \tag{4.109}$$

που δεν ισούται με $y(t-t_0)$ οπότε το σύστημα δεν είναι χρονικά αναλλοίωτο.

Τα συστήματα LTI που είναι γραμμικά και χρονικά αναλλοίωτα χρησιμοποιούνται συχνά στην μελέτη συστημάτων επικοινωνιών. Για παράδειγμα ένα καλώδιο μπορεί να περιγραφεί (τουλάχιστον σε μία πρώτη προσέγγιση) με ένα σύστημα LTI. Πως όμως συνδέεται η είσοδος και η έξοδος σε ένα τέτοιο σύστημα; Μπορεί κανείς να δείξει ότι για κάθε σύστημα LTI υπάρχει μία συνάρτηση $h(t)$ που ονομάζεται *κρουστική απόκριση* που συνδέει την είσοδο με την έξοδο βάσει της παρακάτω σχέσης:

$$y(t) = x(t) * h(t) \tag{4.110}$$

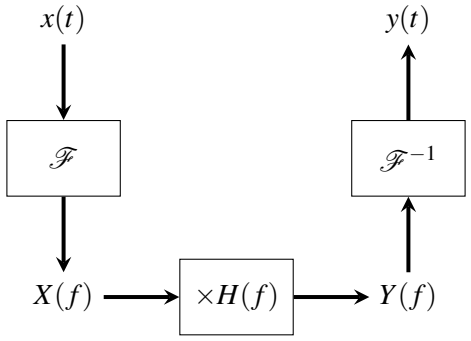
Στην εικόνα 4.23, δείχνουμε την αναπαράσταση ενός συστήματος LTI.

Είναι συχνά πιο εύκολο να περιγράφουμε τα συστήματα LTI στο πεδίο των συχνοτήτων. Αντί να περιγράψουμε την σχέση μεταξύ του σήματος εισόδου $x(t)$ και το σήμα εξόδου $y(t)$ περιγράφουμε την σχέση μεταξύ του φάσματος εισόδου $X(f)$ και το φάσματος εξόδου $Y(f)$ που καθορίζεται ως εξής:

$$Y(f) = H(f)X(f) \tag{4.111}$$

όπου το $H(f)$ είναι ο μετασχηματισμός Fourier του $h(t)$ και ονομάζεται *συνάρτηση μεταφοράς* του συστήματος. Παρατηρούμε ότι η περιγραφή στο πεδίο των συχνοτήτων είναι πολύ απλή εφόσον προκύπτει από τον πολλαπλασιασμό του φάσματος εισόδου $X(f)$ με την συνάρτηση μεταφοράς $H(f)$.

Η διαδικασία φαίνεται στην εικόνα 4.24. Ξεκινώντας από το σήμα εισόδου $x(t)$ υπολογίζουμε τον μετασχηματισμό Fourier $X(f) = \mathscr{F}\{x(t)\}$ και τον πολλαπλασιάζουμε με το $H(f)$ για



Εικόνα 4.24: Περιγραφή ενός LTI συστήματος με χρήση του πεδίου των συχνοτήτων.

να πάρουμε το φάσμα εξόδου $Y(f)$. Στη συνέχεια υπολογίζουμε το $y(t)$ εφαρμόζοντας τον αντίστροφο μετασχηματισμό Fourier $y(t) = \mathscr{F}^{-1}\{Y(f)\}$.

Θα επιλέξουμε να υλοποιήσουμε την κλάση που περιγράφει τα συστήματα LTI, `ltisystem` χρησιμοποιώντας την λογική της εικόνας 4.24 και όχι τον συγκερασμό της εικόνας 4.23. Μαθηματικά είναι ισοδύναμες περιγραφές αλλά η πρώτη περιγραφή έχει το πλεονέκτημα ότι μας δείχνει την μεγάλη σημασία που έχει το πεδίο των συχνοτήτων σε αυτές τις περιπτώσεις καθώς είναι πολύ πιο εύκολο να καταλάβουμε την επίδραση του συστήματος στο πεδίο των συχνοτήτων όπου απλά γίνεται πολλαπλασιασμός του φάσματος $X(f)$ με την συνάρτηση μεταφοράς $H(f)$.

Αρχικά κάνουμε μία αλλαγή στην `__init__` της κλάσης `signal` έτσι ώστε να μπορούμε όταν δημιουργούμε το σήμα να καθορίζουμε και το φάσμα του (δηλαδή το χαρακτηριστικό `spec`). Στο listing 4.19, δείχνουμε την νέα έκδοση της `__init__`. Θα δούμε και παρακάτω πως την χρησιμοποιούμε.

```
68      def __init__(self, t = None, samples = None, f = None, spec = None):
69          """
70          Initialize the signal class
71          """
72          self.t = t
73          if self.t is not None:
74              self.Dt = t[1] - t[0]
75              self.N = t.size
76              self.T = np.max(t) - np.min(t)
77
78          else:
79              self.N = 0
80              self.Dt = None
81
82          self.samples = samples
83          self.f = f
84          self.spec = spec
85
86          if self.f is not None:
87              self.Df = f[1] - f[0]
88              self.N = f.size
89          else:
90              self.Df = None
```

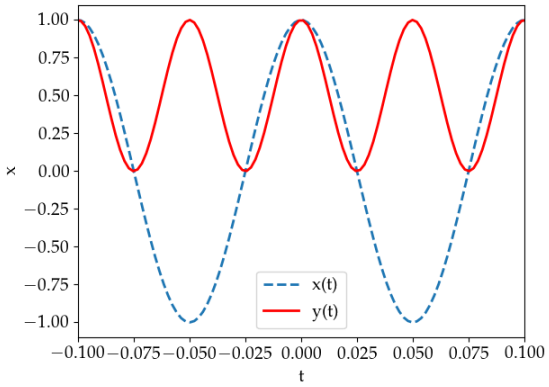
Listing 4.19: `signal.__init__`

Στη συνέχεια φτιάχνουμε μία γενική κλάση `system` η όποια φαίνεται στο listing 4.20 και περιγράφει ένα γενικό σύστημα και η οποία περιέχει μερικές γενικές μεθόδους. Η συμπεριφορά του συστήματος καθορίζεται από την μέθοδο `action` η οποία υπολογίζει την έξοδο από την είσοδο λαμβάνοντας υπόψη και μερικές παραμέτρους που καθορίζονται από το χαρακτηριστικό `params`.

```
225 class system:
226
227     def __init__(self, input_s = None,
228                        output_s = None,
229                        action = None,
230                        params = None):
231
232         self.input_s = input_s
233         self.output_s = output_s
234         self.action = action
235         self.params = params
236
237     def set_input(self, input_s):
238         self.input_s = input_s
239
240     def set_output(self, output_s):
241         self.output_s = output_s
242
243     def set_action(self, action):
244         self.action = action
245
246     def calc_output(self):
247         if self.params is None:
248             self.output_s = self.action( self.input_s )
249         else:
250             self.output_s = self.action( self.input_s, **self.params)
251
252     def calc_spec_input(self):
253         self.input_s.calc_spectrum()
254
255     def calc_spec_output(self):
256         self.output_s.calc_spectrum()
257
258     def calc_inv_input(self):
259         self.input_s.calc_inv_spectrum()
260
261     def calc_inv_output(self):
262         self.output_s.calc_inv_spectrum()
```

Listing 4.20: Η κλάση `system`

```
1 import commlib as cl
2 import numpy as np
3 import matplotlib.pyplot as plt
4
5 def pow2(x):
6     return cl.signal(samples = x.samples ** 2.0, t = x.t)
7
8 Tmin = -1
9 Tmax = 1
```

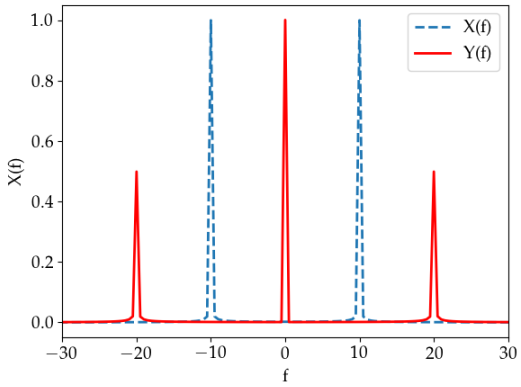


Εικόνα 4.25: Η έξοδος ενός συστήματος με $y(t) = \mathscr{T}\{x(t)\} = x^2(t)$ όταν η είσοδος $x(t)$ είναι ένα φέρον σήμα με πλάτος $A = 1$ και συχνότητα $f_0 = 10$.

```
N = 1024
t = np.linspace(Tmin, Tmax, N)
f0 = 10
x = cl.carrier(t, f0)
x.calc_spectrum()

plt.close('all')


S = cl.system(input_s = x, action = pow2)
S.calc_output()
y = S.output_s
y.calc_spectrum()

plt.figure()
x.plot(line_type = '--')
y.plot(line_type = 'r-')
plt.xlim([-0.1, 0.1])
plt.legend(['x(t)', 'y(t)'])

plt.figure()
plt.plot(x.f, np.abs(x.spec), '--')
plt.plot(y.f, np.abs(y.spec), 'r-')

plt.xlabel('f')
plt.ylabel('X(f)')
plt.legend(['X(f)','Y(f)'])
plt.xlim([-30, 30])
```

Listing 4.21: `systemtest.py`

Στο listing 4.21 δείχνουμε ένα παράδειγμα για την χρήση της κλάσης `system`. Δημιουργούμε ένα φέρον σήμα $x(t) = A\cos(2\pi f_0 t)$ και στη συνέχεια το περνάμε από το σύστημα $y(t) = \mathscr{T}\{x(t)\} = x^2(t)$. Κάνουμε και την γραφική παράσταση της εισόδου και της εξόδου τόσο στο πεδίο του χρόνου όσο και στο πεδίο των συχνοτήτων στην εικόνα 4.25 και 4.26 αντίστοιχα.



Εικόνα 4.26: Το φάσμα εισόδου και εξόδου στην περίπτωση των σημάτων της εικόνας 4.25.

Παρατηρούμε ότι το αρχικό σήμα περιέχει δύο συνιστώσες στις συχνότητες στο $f = \pm f_0 = \pm 10$ αλλά το φάσμα του τελικού σήματος έχει μία συνιστώσα στο $f = 0$ και δύο συνιστώσες στις συχνότητες $f = \pm 2f_0 = \pm 20$ που είναι διπλάσιες των αρχικών. Οι συχνότητες που είναι ακέραιο πολλαπλάσιο των αρχικών ονομάζονται *αρμονικές*. Επομένως στην έξοδο του μη γραμμικού συστήματος παρατηρούμε ότι γεννούνται αρμονικές συχνότητες. Μπορούμε να εξηγήσουμε και θεωρητικά γιατί συμβαίνει αυτό. Δεδομένου ότι $y(t) = x^2(t)$ θα έχουμε:

$$y(t) = x^2(t) = A^2 \cos^2(2\pi f_0 t) = \frac{A^2}{2}\left(1 + \cos(4\pi f_0 t)\right) \tag{4.112}$$

Στην παραπάνω σχέση χρησιμοποιήσαμε την γνωστή τριγωνομετρική σχέση $\cos^2\phi = \frac{1}{2}\left(1 + \cos(2\phi)\right)$. Επομένως ο μετασχηματισμός Fourier του $y(t)$ θα είναι:

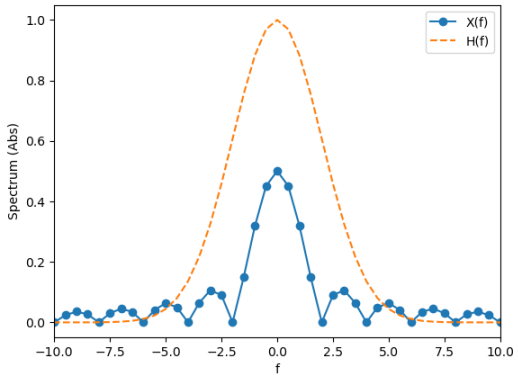
$$Y(f) = \frac{A^2}{4}\delta(f - 2f_0) + \frac{A^2}{4}\delta(f + 2f_0) + \frac{A^2}{2}\delta(f) \tag{4.113}$$

Στην ουσία χρησιμοποιούμε το παράδειγμα 4.6.2 για να υπολογίσουμε το $Y(f)$. Οι συνιστώσες $\frac{A^2}{4}\delta(f \pm 2f_0)$ οφείλονται στον όρο $\frac{A^2}{2}\cos(4\pi f_0 t)$ στην (4.112) ενώ η $\frac{A^2}{2}\delta(f)$ προκύπτει από τον σταθερό όρο $\frac{1}{2}A^2$ ο οποίος μπορεί να θεωρηθεί ότι είναι και αυτός ένα συνημίτονο με συχνότητα $f = 0$, $\frac{1}{2}A^2 = \frac{1}{2}A^2\cos(2\pi 0 t)$) οπότε ο μετασχηματισμός Fourier θα είναι $\frac{1}{2}A^2\delta(f)$.

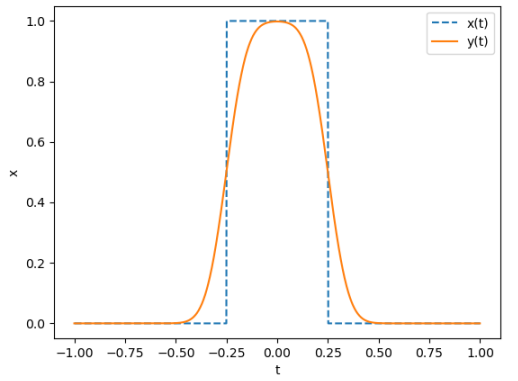
Ας δούμε τώρα ένα παράδειγμα με ένα σύστημα LTI. Στο listing 4.22 εξετάζουμε τι συμβαίνει όταν ένας τετραγωνικός παλμός περνάει από ένα σύστημα LTI με συνάρτηση μεταφοράς:

$$H(f) = \exp\left(-\frac{f^2}{2B^2}\right) \tag{4.114}$$

Η παραπάνω συνάρτηση ονομάζεται Gaussian και την είδαμε ήδη στην (3.35) και θα την χρησιμοποιήσουμε και στα επόμενα κεφάλαια αρκετά συχνά. Έχουμε κάνει μία γραφική παράσταση της συνάρτησης μεταφοράς στην εικόνα 4.27 (με διακεκομμένες γραμμές). Όπως φαίνεται και στην εικόνα η $H(f)$ είναι μέγιστη στο $f = 0$ όπου έχει και την τιμή 1 και για $f > 0$, όσο μεγαλώνει το f τόσο μικραίνει η $H(f)$. Επίσης η $H(f)$ είναι συμμετρική γύρω από το $f = 0$, δηλαδή $H(f) = H(-f)$. Την συγκεκριμένη συμπεριφορά την συναντάμε πολύ συχνά και στην



Εικόνα 4.27: Το φάσμα εισόδου και η συνάρτηση μεταφοράς του συστήματος LTI.



Εικόνα 4.28: Το σήμα εισόδου και εξόδου του συστήματος LTI.

πράξη. Συχνά για παράδειγμα η απόκριση των ηλεκτρονικών κυκλωμάτων χειροτερεύει όσο αυξάνει η συχνότητα f οπότε έχουμε μία αντίστοιχη εξάρτηση. Επίσης πολλά καλώδια και ασύρματα κανάλια έχουν μία παρόμοια συμπεριφορά με αυτή της $H(f)$ στην εικόνα 4.27.

Η παράμετρος B στην (4.114) εκφράζει πόσο γρήγορα πέφτει το πλάτος της $H(f)$ όσο αυξάνει η συχνότητα f. Για $f = B$ έχουμε $H(B) = \exp(-0.5) = 0.606$ οπότε στην συχνότητα αυτή η συνάρτηση μεταφοράς έχει πέσει στο 60.6% της μέγιστης τιμής της. Επομένως όσο μεγαλύτερο είναι το B τόσο καλύτερη είναι η απόκριση του συστήματος στις υψηλότερες συχνότητες.

Θεωρούμε ότι $B = 2$ και πως στην είσοδο του συστήματος έχουμε έναν τετραγωνικό παλμό όπως αυτός του παραδείγματος 4.3.1 με $T_1 = 0.5$. Στην εικόνα 4.27 δείχνουμε με κυκλάκια το φάσμα του σήματος εισόδου $|X(f)|$, που είναι μία συνάρτηση sinc όπως άλλωστε είδαμε και στο παράδειγμα. Παρατηρείστε ότι η συνάρτηση $H(f)$ περιλαμβάνει ολόκληρο τον κεντρικό λοβό του $X(f)$ αλλά και ένα μέρος των πλευρικών λοβών στις υψηλότερες συχνότητες. Ωστόσο για $f = 3B$ θα έχουμε $H(3B) = \exp(-4.5) = 0.0111$ οπότε έξω από το διάστημα $f \in [-3B, 3B]$ το πλάτος της συνάρτησης $H(f)$ είναι μικρότερο από $\approx 1.1\%$ της μέγιστης τιμής του και επομένως

οι συνιστώσες του φάσματος εξόδου $Y(f) = H(f)X(f)$, στις συχνότητες αυτές θα είναι πολύ εξασθενημένες για $|f| > 3B$. Στην εικόνα 4.28 δείχνουμε τι συμβαίνει στο πεδίο του χρόνου. Ο παλμός εισόδου $x(t)$ είναι με διακεκομμένες γραμμές και είναι τετραγωνικός, ωστόσο ο παλμός εισόδου παρουσιάζει διαφορετική συμπεριφορά. Επειδή οι υψηλές συχνότητες του σήματος έχουν εξασθενίσει εξαιτίας της $H(f)$, ο παλμός περιέχει πιο αργές μεταβάσεις από την μηδενική στάθμη στην μέγιστη τιμή του και το ανάποδο. Ο παλμός εξόδου $y(t)$ είναι πιο διευρυμένος από τον παλμό εισόδου $x(t)$.

Φανταστείτε επομένως τι συμβαίνει όταν προσπαθούμε να περάσουμε τέτοιου είδους σήματα από συστήματα που δεν μπορούν να "ακούσουν" τις υψηλές συχνότητες. Οι παλμοί εισόδου εμφανίζονται διευρυμένοι στην έξοδο όπως φαίνεται και στην εικόνα 4.28. Όπως θα δούμε και παρακάτω, συνήθως κωδικοποιούμε την πληροφορία χρησιμοποιώντας διαδοχικούς παλμούς και επομένως υπάρχει κίνδυνος ένας παλμός να διευρυνθεί σημαντικά και ένα μεγάλο του μέρος να πέσει μέσα στην διάρκεια του επόμενου παλμού προκαλώντας παραμόρφωση παρεμβολής. Αυτό το φαινόμενο αναφέρεται στην βιβλιογραφία ως *αλληλοπαρεμβολή συμβόλων* (intersymbol interference - ISI) και αποτελεί σημαντικό πρόβλημα στα περισσότερα συστήματα επικοινωνιών υψηλών ταχυτήτων. Συχνά, πρέπει να υλοποιηθούν ειδικές μέθοδοι για την αντιμετώπιση του και ένας από τους λόγους της επιτυχίας των τεχνολογιών ψηφιακής συνδρομητικής γραμμής (digital subscriber line - DSL) είναι το γεγονός ότι χρησιμοποιούν προηγμένες τεχνικές αντιστάθμισης της ISI.

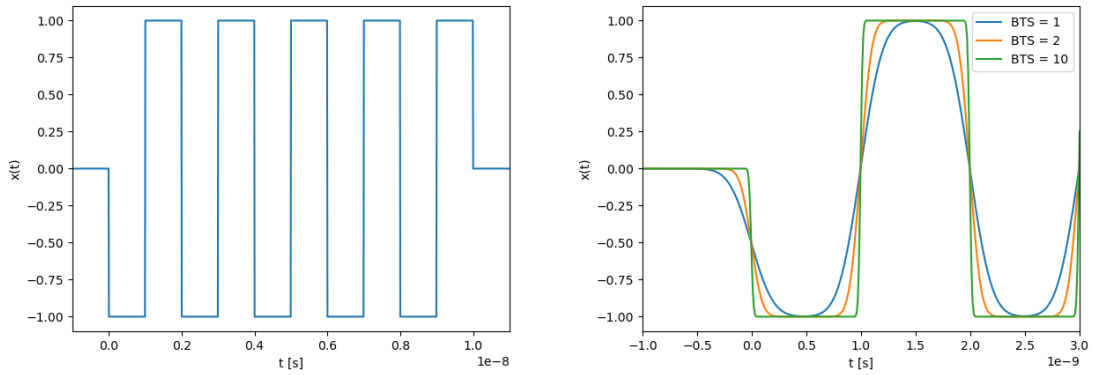
```
import commlib as cl
import numpy as np
import matplotlib.pyplot as plt

Tmin = -1
Tmax = 1
T1 = 0.5
N = 1024
t = np.linspace(Tmin, Tmax, N)

x = cl.square_pulse(t, T1)
x.calc_spectrum()

plt.close('all')



B = 2
H = lambda f: np.exp( -f **2.0 / 2 / B ** 2.0)

plt.figure()
plt.plot(x.f, np.abs(x.spec), '-o', label = 'X(f)' )
plt.plot(x.f, H(x.f), '--', label = 'H(f)' )
plt.xlabel('f')
plt.ylabel('Spectrum (Abs)')
plt.legend()
plt.xlim([-10, 10])

S = cl.lti_system(input_s = x, H = H)
S.calc_output()
y = S.output_s
plt.figure()
x.plot(line_type = '--', label = 'x(t)')
```

```
y.plot(line_type = '-', label = 'y(t)')
plt.legend()
```

Listing 4.22: Παράδειγμα συστήματος LTI

Ας δούμε λίγο παραπάνω την επίδραση της $H(f)$ σε ένα ψηφιακό σήμα. Χρησιμοποιούμε το listing 4.23 για να φτιάξουμε ένα ψηφιακό σήμα το οποίο το περνάμε από ένα LTI σύστημα με $H(f)$ που δίνεται από την (4.114). Στην εικόνα 4.29a δείχνουμε το σήμα εισόδου που υλοποιούμε στο listing 4.23 που αποτελείται από 8 διαδοχικούς παλμούς που εναλλάσσονται σε πλάτος ± 1. Η διάρκεια συμβόλου τίθεται ίση με $T_S = 1$ ns. Το σήμα περνάει από το LTI σύστημα και στην εικόνα 4.29b δείχνουμε το σήμα εξόδου για διαφορετικές τιμές του γινομένου BT_S. Για $BT_S = 10$, περιμένουμε ότι το σύστημα έχει πολύ καλή απόκριση μέσα στη συχνοτική περιοχή του σήματος εισόδου οπότε θα έχουμε και ελάχιστη παραμόρφωση στην έξοδο. Όσο όμως μικραίνει το BT_S (οπότε ελαττώνεται το B) η αλλοίωση είναι πιο εμφανής καθώς οι παλμοί παρουσιάζονται διευρυμένοι στην έξοδο.

```
from commlib import digital_signal, lti_system
import numpy as np
import matplotlib.pyplot as plt

bs = np.array([1, 2, 10])

xi = np.array([-1, 1, -1, 1, -1, 1, -1, 1, -1, 1])
TS = 1.0e-9
tguard = 20.0 * TS
tinitial = 0.0
samples_per_symbol = 100
s = digital_signal(TS = TS,
                   samples_per_symbol = samples_per_symbol,
                   tinitial = tinitial,
                   tguard = tguard)
plt.close('all')
s.modulate_from_symbols(xi)

plt.close('all')
plt.figure(1)
s.plot(line_type = '-')
plt.xlim([-TS, (xi.size + 1) * TS])
plt.xlabel('t [s]')
plt.ylabel('x(t)')

plt.figure(2)
for b in bs:
    B = b / TS
    H = lambda f: np.exp( -f **2.0 / 2 / B ** 2.0)
    L = lti_system(H = H, input_s = s)
    L.calc_output()
    sout = L.output_s
    sout.plot(line_type = '-',
              label = 'BTS = ' + str(b))

plt.xlabel('t [s]')
plt.ylabel('x(t)')
plt.legend()
```



(a) (b)

Εικόνα 4.29: (a) Το σήμα εισόδου στο σύστημα LTI και (b) η έξοδος του συστήματος

```
39 plt.xlim([-TS, 3 * TS])
```

Listing 4.23: `ltitestdig.py`

4.11 Τι μάθαμε

Το συγκεκριμένο κεφάλαιο κάναμε μία εισαγωγή στο πεδίο των συχνοτήτων που παίζει πολύ σημαντικό ρόλο στην κατανόηση των συστημάτων επικοινωνιών. Ασχοληθήκαμε με την σειρά Fourier που χρησιμοποιείται για σήματα πεπερασμένης διάρκειας και περιοδικά σήματα. Επίσης είδαμε τον μετασχηματισμό Fourier που μπορεί να χρησιμοποιηθεί για γενικά σήματα τα οποία δεν είναι απαραίτητα πεπερασμένης διάρκειας ή περιοδικά. Το πεδίο των συχνοτήτων μας δίνει έναν εύκολο τρόπο να περιγράφουμε τα συστήματα που χρησιμοποιούμε στις επικοινωνίες (π.χ. ενισχυτές κτλ) αλλά και κανάλια επικοινωνιών (καλώδια και ασύρματα κανάλια).

5. Τυχαία Σήματα

5.1 Εισαγωγή

Οι επιδόσεις ενός συστήματος επικοινωνιών περιορίζονται από τον θόρυβο. Με τον όρο *θόρυβο* εννοούμε οποιοδήποτε σήμα $n(t)$ το οποίο παρεμβάλλεται με το αρχικό σήμα $x(t)$. Το σήμα θορύβου $n(t)$ είναι *τυχαίο* και επομένως δεν μπορούμε να προβλέψουμε την τιμή του και επομένως να το απομακρύνουμε. Ένα παράδειγμα θορύβου είναι ο *προσθετικός* θόρυβος ο οποίος προστίθεται στο σήμα, οπότε το τελικό σήμα στον δέκτη $y(t)$ δίνεται από την σχέση:

$$y(t) = x(t) + n(t) \tag{5.1}$$

Υπάρχουν και άλλα είδη θορύβου. Για παράδειγμα ο θόρυβος *φάσης* $\Delta\phi(t)$ προστίθεται στη φάση του σήματος και όχι απευθείας στο πλάτος του. Αν έχουμε ένα αρμονικό σήμα:

$$x(t) = A\cos(2\pi f_0 t) \tag{5.2}$$

τότε στο τελικό σήμα, ο θόρυβος φάσης επιδρά ως εξής:

$$y(t) = A\cos(2\pi f_0 t + \Delta\phi(t)) \tag{5.3}$$

Ο θόρυβος έχει πολλές αιτίες. Ο *θερμικός* θόρυβος που ονομάζεται και θόρυβος Johnson-Nyquist οφείλεται στην τυχαία κίνηση των ηλεκτρονίων στα ηλεκτρονικά κυκλώματα και υπάρχει ανεξάρτητα από το αν στο κύκλωμα εφαρμόζουμε κάποια τάση. Ο θόρυβος *βολής* είναι ένα άλλο παράδειγμα και εμφανίζεται εξαιτίας του γεγονότος ότι τα ηλεκτρόνια που αποτελούν το ηλεκτρικό ρεύμα κινούνται (περίπου) ως σωματίδια και έχουν ένα πολύ μικρό αλλά μετρήσιμο ηλεκτρικό φορτίο q_e. Επομένως το ηλεκτρικό ρεύμα που περνάει από την διατομή ενός αγωγού, θα παρουσιάζει διακυμάνσεις που οφείλονται στην τυχαιότητα άφιξης των ηλεκτρονίων. Για μεγάλα ρεύματα, οι διακυμάνσεις αυτές θα είναι ελάχιστες αλλά όσο μικραίνουμε την τιμή του ρεύματος, θα γίνονται πιο σημαντικές. Και οι δύο θόρυβοι είναι προσθετικοί.

Στο κεφάλαιο αυτό θα επικεντρωθούμε στις ιδιότητες του θορύβου οι οποίες καθορίζουν και τις επιδόσεις ενός συστήματος επικοινωνιών. Δεδομένου ότι ο θόρυβος είναι τυχαίος θα

ξεκινήσουμε με κάποια στοιχεία από την θεωρία πιθανοτήτων που θα είναι χρήσιμα για την περιγραφή των *τυχαίων* σημάτων. Στη συνέχεια θα εξετάσουμε τις ιδιότητες του θορύβου τόσο στο πεδίο του χρόνου όσο και στο πεδίο των συχνοτήτων.

5.2 Τυχαίες Μεταβλητές

Υπάρχουν περιπτώσεις όπου δεν μπορούμε να προβλέψουμε την εξέλιξη ενός φαινομένου ή ενός πειράματος αλλά έχουμε την δυνατότητα να υπολογίσουμε την *πιθανότητα* των διάφορων αποτελεσμάτων. Έστω για παράδειγμα ότι ρίχνουμε ένα ζάρι N φορές και συμβολίζουμε με D το αποτέλεσμα έτσι ώστε $D = 1$ όταν έρθει άσος, $D = 2$ όταν έρθουν διπλές κ.ο.κ. Αν το ζάρι είναι τίμιο τότε θα πρέπει αν το N είναι μεγάλο, σχεδόν $N_1 = N/6$ φορές να έρθει $D = 1$, $N_2 = N/6$ φορές να έρθει $D = 2$ κτλ. Η πιθανότητα $\Pr\{D = i\}$ να έχουμε $D = i$ για $1 \leq i \leq 6$ ορίζεται ως το πλήθος των φορών N_i που έχουμε το αποτέλεσμα i ως προς τον συνολικό αριθμό N που εκτελούμε το πείραμα, όταν το N είναι πολύ μεγάλο (θεωρητικά άπειρο):

$$\Pr\{D = i\} = \lim_{N\to\infty} \frac{N_i}{N} \tag{5.4}$$

Στην περίπτωση του ζαριού θα έχουμε:

$$\Pr\{D = i\} = \frac{1}{6} \tag{5.5}$$

Το D υπολογίζεται βάσει ενός τυχαίου πειράματος και ονομάζεται επομένως *τυχαία μεταβλητή*. Αν συμβολίσουμε με $\{D_1, D_2, \cdots, D_N\}$ τις διαδοχικές τιμές που λαμβάνουμε εκτελώντας το πείραμα τότε τα D_k για $1 \leq k \leq N$ μπορούν να παίρνουν τις διακριτές τιμές $\{1,2,3,4,5,6\}$ και μόνο αυτές. Ονομάζουμε τις μεταβλητές αυτές *διακριτές* τυχαίες μεταβλητές.

Υπάρχουν και περιπτώσεις όπου μία τυχαία μεταβλητή μπορεί να λάβει οποιαδήποτε τιμή μέσα σε ένα διάστημα τιμών $[a,b]$, οπότε την ονομάζουμε *συνεχή* τυχαία μεταβλητή. Ας υποθέσουμε για παράδειγμα ότι παρατηρούμε την έξοδο μίας γεννήτριας εναλλασσόμενης τάσης την οποία κάποιος ανάβει μία τυχαία χρονική στιγμή οπότε η τάση εξόδου v_{OUT} θα δίνεται από μία σχέση της μορφής:

$$v_{\text{OUT}}(t) = V_0 \cos(2\pi f_0 t + \phi) \tag{5.6}$$

όπου απλά έχουμε αντικαταστήσει το ϕ είναι μία τυχαία αρχική φάση στο διάστημα $[0, 2\pi]$. Το ϕ είναι μία συνεχής τυχαία μεταβλητή που λαμβάνει τιμές στο διάστημα $[0, 2\pi]$.

Ας θεωρήσουμε ένα δεύτερο παράδειγμα, όπου μετράμε το πλάτος της τάσης V σε μία αντίσταση R την οποία δεν έχουμε συνδέσει σε κάποια πηγή τάσης. Εξαιτίας του θερμικού θορύβου, η τάση θα παρουσιάζει τυχαίες διακυμάνσεις και επομένως το V είναι τυχαία μεταβλητή. Από την θεωρία του θερμικού θορύβου προκύπτει ότι το διάστημα πιθανών τιμών είναι το $(-\infty, +\infty)$ και πως το V είναι τυχαία μεταβλητή.

Για τις συνεχείς τυχαίες μεταβλητές δεν έχει νόημα να υπολογίζουμε την πιθανότητα η μεταβλητή να πάρει μία συγκεκριμένη τιμή. Για παράδειγμα, αν υπολογίζαμε την πιθανότητα $\Pr\{\phi = \pi\}$ για την τυχαία μεταβλητή ϕ στην (5.6) θα χρησιμοποιούσαμε μία σχέση παρόμοια με την (5.4),

$$\Pr\left\{\phi = \frac{\pi}{2}\right\} = \lim_{N\to\infty} \frac{N_{\pi/2}}{N} \tag{5.7}$$

όπου $N_{\pi/2}$ είναι ο αριθμός των φορών όπου πετυχαίνουμε ακριβώς την τιμή $\pi/2$ όταν κάνουμε N φορές το πείραμα. Δεδομένου όμως ότι υπάρχουν άπειρες τιμές μέσα στο διάστημα $[0, 2\pi]$, περιμένουμε ότι το $N_{\pi/2}$ θα είναι πάρα πολύ μικρό σε σχέση με το N οπότε,

$$\Pr\left\{\phi = \frac{\pi}{2}\right\} = \lim_{N\to\infty} \frac{N_{\pi/2}}{N} = 0 \tag{5.8}$$

Άρα για μία συνεχή τυχαία μεταβλητή δεν έχει νόημα να ασχολούμαστε με την πιθανότητα να λάβει μία συγκεκριμένη τιμή καθώς αυτή θα είναι μηδενική. Για μία συνεχή μεταβλητή μας ενδιαφέρει περισσότερο η πιθανότητα να βρεθεί μέσα σε ένα διάστημα τιμών $[\gamma, \delta]$, π.χ.

$$\Pr\{\gamma \leq \phi \leq \delta\} \tag{5.9}$$

Συνχά μπορούμε να υπολογίσουμε την *πυκνότητα πιθανότητας* (probability density function - PDF) μίας τυχαίας μεταβλητής $f_\phi(x)$ από την οποία υπολογίζονται οι πιθανότητες (5.9) ολοκληρώνοντας την $f_\phi(x)$ στο εν λόγω διάστημα, δηλαδή:

$$\Pr\{\gamma \leq \phi \leq \delta\} = \int_\gamma^\delta f_\phi(x)\mathrm{d}x \tag{5.10}$$

Ορίζουμε επίσης την αθροιστική πυκνότητα πιθανότητας $F_\phi(y)$ (cumulative distribution function - CDF) ως την πιθανότητα η τυχαία μεταβλητή ϕ να λάβει τιμή μικρότερη ή ίση από το y, δηλαδή:

$$F_\phi(y) = \Pr\{\phi \leq y\} = \int_{-\infty}^{y} f_\phi(x)\mathrm{d}x \tag{5.11}$$

Από την (5.11) μπορεί να προκύψει ένας ενδιαφέρον ορισμός της PDF. Αν παραγωγίσουμε την (5.11) ως προς y, θα έχουμε:

$$\frac{\mathrm{d}}{\mathrm{d}y}F_\phi(y) = \frac{\mathrm{d}}{\mathrm{d}y}\int_{-\infty}^{y} f_\phi(x)\mathrm{d}x = f_\phi(y) \tag{5.12}$$

Σύμφωνα με την (5.12) η PDF είναι η παράγωγος της CDF. Στο όριο όπου το y πλησιάζει στο $+\infty$ θα πρέπει να έχουμε

$$F_\phi(+\infty) = \Pr\{\phi < +\infty\} = 1 \tag{5.13}$$

Η παραπάνω εξίσωση εκφράζει το γεγονός ότι σίγουρα (δηλαδή με πιθανότητα 1), το ϕ θα είναι μικρότερο του $+\infty$. Χρησιμοποιώντας και την (5.11) προκύπτει εύκολα ότι:

$$F_\phi(+\infty) = \int_{-\infty}^{+\infty} f_\phi(x)\mathrm{d}x = 1 \tag{5.14}$$

Επομένως το ολοκλήρωμα της πυκνότητας πιθανότητας κατά μήκος του πραγματικού άξονα θα πρέπει να ισούται με ένα. Σε πολλές περιπτώσεις η PDF μπορεί να υπολογιστεί βάσει της θεωρίας ή μέσω προσομοιώσεων. Ας δούμε μερικά παραδείγματα.

5.2.1 Ομοιόμορφη Τυχαία Μεταβλητή

Η τυχαία μεταβλητή X λέμε ότι ακολουθεί *ομοιόμορφη* κατανομή σε ένα διάστημα $[a,b]$ όταν έχει PDF:

$$f_X(x) = \begin{cases} \frac{1}{b-a} & ,a \leq x \leq b \\ 0 & ,\text{διαφορετικά} \end{cases} \tag{5.15}$$

Η CDF $F_X(y)$ της ομοιόμορφης κατανομής υπολογίζεται από την (5.12),

$$F_X(y) = \int_{-\infty}^{y} f_X(x)\mathrm{d}x \tag{5.16}$$

Αν $y < a$ τότε θα έχουμε $f_X(x) = 0$ για $-\infty < x < y$ οπότε ολοκληρώνουμε μία συνάρτηση η οποία είναι παντού μηδέν στο διάστημα ολοκλήρωσης της (5.16), οπότε $F_X(y) = 0$. Όταν $a \leq y \leq b$, η συνάρτηση που ολοκληρώνουμε θα είναι σταθερή και ίση με $1/(b-a)$ στο διάστημα $[a,y]$ και μηδέν εκτός αυτού, οπότε:

$$F_X(y) = \int_{-\infty}^{y} f_X(x)\mathrm{d}x = \frac{1}{b-a}\int_{a}^{y} \mathrm{d}x = \frac{y-a}{b-a} \tag{5.17}$$

Όταν $y > b$ θα έχουμε:

$$F_X(y) = \int_{-\infty}^{y} f_X(x)\mathrm{d}x = \frac{1}{b-a}\int_{a}^{b} \mathrm{d}x = 1 \tag{5.18}$$

Συνδυάζοντας τις (5.16), (5.17) και (5.18) μπορούμε να γράψουμε:

$$F_Y(y) = \begin{cases} 0 & ,y < a \\ \frac{y-a}{b-a} & ,a \leq y \leq b \\ 1 & ,y > b \end{cases} \tag{5.19}$$

Μπορούμε να δοκιμάσουμε στην πράξη τις έννοιες που αφορούν την ομοιόμορφη κατανομή χρησιμοποιώντας την `Python`. Η `numpy` έχει μία συνάρτηση την `numpy.random.rand` που παράγει ψευδοτυχαία δείγματα $\bar{x}_i$ βάσει της ομοιόμορφης κατανομής με $a = 0$ και $b = 1$. Στο listing 5.1 δείχνουμε μία προσθήκη στην `commlib`, την συνάρτηση `calc_pdfcdf` που μπορεί να χρησιμοποιηθεί για να υπολογίσει την PDF και την CDF στις τιμές που καθορίζονται από τον πίνακα `y` και που αντιστοιχούν στα δείγματα του πίνακα `x`. Η λογική πίσω από τον υπολογισμό της PDF βασίζεται στην (5.12), όπου για μικρό Δy θα έχουμε:

$$\frac{F_X(y+\frac{1}{2}\Delta y) - F_X(y-\frac{1}{2}\Delta y)}{\Delta y} \approx f_X(y) \tag{5.20}$$

Ο αριθμητής μπορεί να γραφτεί και ως εξής:

$$\begin{aligned} F_X(y+\tfrac{1}{2}\Delta y) - F_X(y-\tfrac{1}{2}\Delta y) &= \int_{-\infty}^{y+\frac{1}{2}\Delta y} f_X(x)\mathrm{d}x - \int_{-\infty}^{y-\frac{1}{2}\Delta y} f_X(x)\mathrm{d}x \\ &= \int_{y-\frac{1}{2}\Delta y}^{y+\frac{1}{2}\Delta y} f_X(x)\mathrm{d}x = \Pr\left\{y-\tfrac{1}{2}\Delta y \leq X \leq y+\tfrac{1}{2}\Delta y\right\} \end{aligned} \tag{5.21}$$

Συνδυάζοντας τις (5.20) και (5.21) καταλήγουμε στο ότι:

$$f_X(y) \approx \frac{\Pr\left\{y - \frac{1}{2}\Delta y \leq X \leq y + \frac{1}{2}\Delta y\right\}}{\Delta y} \tag{5.22}$$

Στο όριο όπου το Δy γίνεται πολύ μικρό, γράψουμε:

$$f_X(y) = \lim_{\Delta y \to 0} \frac{\Pr\left\{y - \frac{1}{2}\Delta y \leq X \leq y + \frac{1}{2}\Delta y\right\}}{\Delta y} \tag{5.23}$$

```
240 def calc_pdfcdf(x, y):
241     N = x.size
242     Dy = y[1] - y[0]
243     pdf = np.zeros( y.size )
244     cdf = np.zeros( y.size )
245 
246     for i, yy in enumerate(y):
247         Ny = np.sum( (x >= yy-Dy/2) & ( x<yy+Dy/2 ) )
248         pdf[i] =  Ny / N / Dy
249         cdf[i] = np.sum( x <= yy ) / N
250 
251     return pdf, cdf
252 #---calcpdfcdf2
253 
254 #---qfunction1
255 def Q(x):
256     return 0.5 * erfc( x / np.sqrt(2) )
257 
258 def normcdf(x, mu, sigma):
259     z = (x - mu) / sigma
260     return 1 - Q(z)
261 #---qfunction2
262 
263 
264 #---digitalsignal1
265 class digital_signal(signal):
266 
267     def __init__(self, TS = 1e-6, samples_per_symbol = 10,
268                  tinitial = 0, tguard = 0.0):
269 
270         super().__init__()
271         self.TS = TS
272         self.samples_per_symbol = samples_per_symbol
273         self.tinitial = tinitial
274         self.tguard = tguard
275 
276     def modulate_from_symbols( self, symbols ):
277 
278         self.Tmin = self.tinitial - self.tguard
279         self.Tmax = self.tinitial + symbols.size * self.TS +  self.tguard
280         self.Dt = self.TS / self.samples_per_symbol
281         self.t = np.arange(self.Tmin, self.Tmax, self.Dt)
282         self.samples = np.zeros( self.t.size )
283         self.symbols = symbols
284         self.N = self.t.size
```

```
285
286         i = np.floor( (self.t - self.tinitial) / self.TS).astype(int)
287         j = np.where( np.logical_and(i >= 0, i < symbols.size ) )
288
289         self.samples[j] = symbols[ i[j] ]
290 #---digitalsignal2
```

Listing 5.1: Η συνάρτηση calc_pdfcdf

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3  from commlib import calc_pdfcdf
4
5  Nsamples = 100000
6  a = 1
7  b = 2
8  Nplot = 1000
9
10
11 x = a + (b-a) * np.random.rand(Nsamples)
12 plt.close('all')
13 plt.plot(x[:Nplot],'s')
14 plt.xlabel('Sample Index')
15 plt.ylabel('Sample Value')
16
17 y = np.linspace(a-1, b+1, 100)
18 pdf, cdf = calc_pdfcdf(x, y)
19
20 plt.figure()
21 plt.plot(y, pdf)
22 plt.xlabel('y')
23 plt.ylabel('PDF')
24 plt.title('PDF of Uniform Samples')
25
26 plt.figure()
27 plt.plot(y, cdf)
28 plt.xlabel('y')
29 plt.ylabel('CDF')
30 plt.title('CDF of Uniform Samples')
```

Listing 5.2: `uniform.py`

Η `calc_pdfcdf` στο listing 5.1 προσεγγίζει την PDF σύμφωνα με την (5.22). Μετράμε τον αριθμό $N(y)$ των δειγμάτων X τα οποία βρίσκονται μεταξύ $y-\frac{1}{2}\Delta y \leq X \leq y+\frac{1}{2}\Delta y$ με την εντολή:

```
Ny = np.sum( (x >= yy-Dy/2) & ( x<yy+Dy/2 ) )
```

Προσέξτε ότι ο πίνακας:

```
  (x >= yy-Dy/2) & ( x<yy+Dy/2 )
```

είναι ίσος με `True` για όσα δείγματα του πίνακα `x` είναι μεγαλύτερα από `yy-Dy/2` και μικρότερα από `yy+Dy/2` και `False` διαφορετικά. Όταν εφαρμόζεται η `numpy.sum` τότε τα `True` μετατρέπονται σε `1` και τα `False` σε `0` και επομένως το αποτέλεσμα του αθροίσματος ισούται με τον αριθμό των στοιχείων για τα οποία ισχύει η συνθήκη. Στη συνέχεια προσεγγίζουμε την πιθανότητα $\Pr\left\{y-\frac{1}{2}\Delta y \leq X \leq y+\frac{1}{2}\Delta y\right\}$ με το πηλίκο N_y/N, του N_y δια τον συνολικό αριθμό των δειγμάτων N. Σύμφωνα με την (5.22) αν διαιρέσουμε με Δy θα έχουμε μία εκτίμηση της

πυκνότητας πιθανότητας η οποία βέβαια θα βελτιώνεται όσο αυξάνει το πλήθος των δειγμάτων N. Οπότε μαθηματικά γράφουμε:

$$f_X(y) \approx \frac{N_y}{N\Delta y} \tag{5.24}$$

που στον κώδικα αντιστοιχεί σε:

```
pdf[i] =  Ny / N / Dy
```

Παρόμοιος είναι και ο υπολογισμός της CDF. Για κάθε τιμή yy του y υπολογίζουμε το πλήθος των δειγμάτων του x που είναι μικρότερο από yy και διαιρούμε με το πλήθος των δειγμάτων για να εκτιμήσουμε την πιθανότητα:

$$F_X(y) = \Pr\{X \leq y\} \tag{5.25}$$

Το αντίστοιχο τμήμα του κώδικα είναι το εξής:

```
cdf[i] = np.sum( x <= yy ) / N
```

Στην εικόνα 5.1a, δείχνουμε τα πρώτα 1000 δείγματα που παράγονται από την `numpy.random.rand` και τα οποία τα πολλαπλασιάζουμε με $(b-a)$ και προσθέτουμε a. Η `numpy.random.rand` παράγει δείγματα στο διάστημα $[0,1]$ ενώ εμείς στο listing 5.2, θέλουμε τα δείγματα να είναι στο διάστημα $[a,b]$ με $a=1$ και $b=2$. Οπότε κάνουμε αυτήν την μετατροπή:

```
x = a + (b-a) * np.random.rand(Nsamples)
```

για να φέρουμε τα παραγόμενα δείγματα στο σωστό διάστημα. Με τον τρόπο αυτό όπως παρατηρούμε και στην εικόνα, όλα τα δείγματα είναι μεταξύ 1 και 2. Στην εικόνα 5.1b δείχνουμε την PDF που υπολογίζεται από την `calc_pdfcdf`. Όπως περιμένουμε μοιάζει με την PDF μίας ομοιόμορφης κατανομής όπως αυτή καθορίζεται από την (5.15). Υπάρχουν βέβαια κάποιες διακυμάνσεις, οι οποίες αναμένουμε να εξομαλυνθούν αν τρέξουμε πάλι το πείραμα για μεγαλύτερο αριθμό δειγμάτων. Επίσης η μορφή της CDF στην εικόνα 5.1c είναι σύμφωνη με την (5.19): μέχρι το $y=a$ είναι μηδέν, από το $y=b$ και μετά είναι 1 και ενδιάμεσα είναι γραμμική.

5.2.2 Αναμενόμενη τιμή

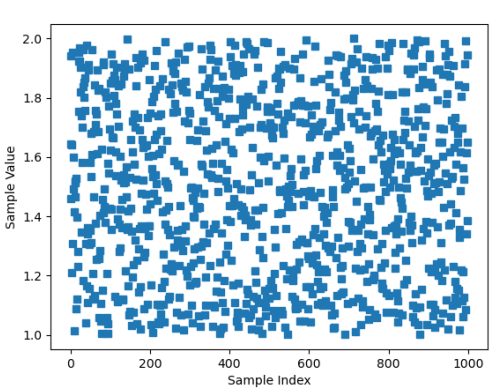
Η *αναμενόμενη τιμή* είναι μία έννοια που θα μας χρησιμεύσει ιδιαίτερα στα παρακάτω κεφάλαια. Η αναμενόμενη τιμή μίας συνάρτησης $g(X)$ μίας τυχαίας μεταβλητής X συμβολίζεται με $\mathbb{E}\{g(X)\}$. Στην περίπτωση μίας διακριτής τυχαίας μεταβλητής ορίζεται ως εξής:

$$\mathbb{E}\{g(X)\} = \sum_i g(x_i)\Pr\{X = x_i\} \tag{5.26}$$

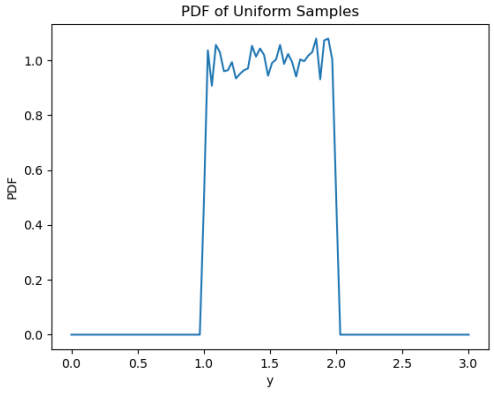
Αν κάνουμε ένα μεγάλο αριθμό πειραμάτων N και παράγουμε δείγματα X_k της X και N_i είναι ο αριθμός των φορών που εμφανίζεται το x_i ως τιμή στα δείγματα αυτά, τότε αν το N είναι πολύ μεγάλο θα έχουμε:

$$\mathbb{E}\{g(X)\} = \sum_i \frac{g(x_i)N_i}{N} = \frac{1}{N}\sum_k g(X_k) \tag{5.27}$$

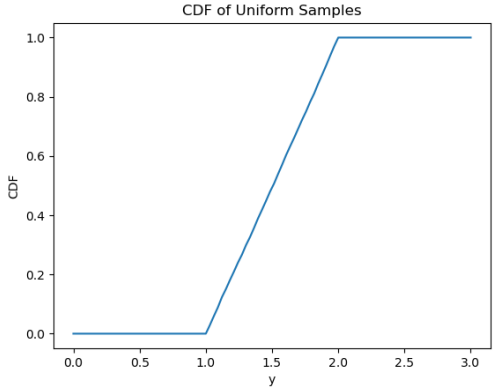
Επομένως η αναμενόμενη τιμή $\mathbb{E}\{g(X)\}$ είναι η *μέση τιμή* των δειγμάτων $Z_k = g(X_k)$ που παράγονται εφαρμόζοντας την συνάρτηση g στα διαδοχικά δείγματα X_k που παράγονται από



(a)



(b)



(c)

Εικόνα 5.1: (a) Δείγματα που παράγονται από την `np.random.rand`, (b) και (c) η PDF και η CDF αντίστοιχα που υπολογίζονται αριθμητικά από την `calc_pdfcdf` .

την X. Αν $g(X) = X$ τότε το $\mu_X = \mathbb{E}\{g(X)\} = \mathbb{E}\{X\}$ ονομάζεται *μέση τιμή* της X. Στην περίπτωση του ζαριού που είδαμε στην ενότητα 5.2, θα έχουμε:

$$\mu_D = \mathbb{E}\{D\} = \frac{1}{6}(1+2+3+4+5+6) = 3.5 \tag{5.28}$$

Οπότε αν ρίξουμε πάρα πολλές φορές το ζάρι και σημειώσουμε τις τιμές του αποτελέσματος και πάρουμε την μέση τιμή όλων αυτών θα λάβουμε μία τιμή πολύ κοντά στο $\mu_D = 3.5$ (θεωρητικά με άπειρο αριθμό πραγματοποιήσεων θα λάβουμε ακριβώς το $\mu_D = 3.5$).

Θα δούμε παρακάτω ότι πέρα από την μέση τιμή $\mu_X = \mathbb{E}\{X\}$ μας ενδιαφέρει και η διακύμανση (variance), σ_X^2, η οποία ορίζεται από την:

$$\sigma_X^2 = \mathbb{E}\left\{(X-\mu_X)^2\right\} \tag{5.29}$$

Στην περίπτωση του ζαριού, αν εφαρμόσουμε την (5.26) θα έχουμε:

$$\begin{aligned}\sigma_D^2 = \mathbb{E}\left\{(D-\mu_D)^2\right\} &= \sum_{i=1}^{6}(i-\mu_D)^2 \Pr\{D=i\} = \frac{1}{6}\sum_{i=1}^{6}(i-3.5)^2 \\ &= \frac{1}{6}\left[(1-3.5)^2+(2-3.5)^2+(3-3.5)^2+(4-3.5)^2+(5-3.5)^2+(6-3.5)^2\right] \\ &= 2.1966\end{aligned} \tag{5.30}$$

Στην περίπτωση των συνεχών τυχαίων μεταβλητών ο υπολογισμός των αναμενόμενων τιμών είναι παρόμοιος με την διαφορά ότι αντί των πιθανοτήτων $\Pr\{X = x_i\}$ χρησιμοποιούμε την πυκνότητα πιθανότητας $f_X(x)$:

$$\mathbb{E}\{g(X)\} = \int_{-\infty}^{+\infty} g(x) f_X(x)\mathrm{d}x \tag{5.31}$$

Ας υπολογίσουμε την μέση τιμή και την διακύμανση στην περίπτωση μίας ομοιόμορφης τυχαίας μεταβλητής. Για την μέση τιμή έχουμε:

$$\mu_X = \int_{-\infty}^{+\infty} x f_X(x)\mathrm{d}x = \int_a^b \frac{1}{b-a} x\mathrm{d}x = \frac{1}{b-a}\int_a^b x\mathrm{d}x = \frac{1}{b-a}\left(\frac{b^2}{2}-\frac{a^2}{2}\right) = \frac{b+a}{2} \tag{5.32}$$

ενώ για την διακύμανση,

$$\begin{aligned}\sigma_X^2 = \mathbb{E}\left\{(X-\mu_X)^2\right\} = \int_{-\infty}^{+\infty}(x-\mu_X)^2 f_X(x)\mathrm{d}x &= \int_a^b \frac{1}{b-a}(x-\mu_X)^2\,\mathrm{d}x \\ &= \frac{1}{b-a}\int_a^b (x-\mu_X)^2\mathrm{d}x = \frac{1}{b-a}\frac{(b-\mu_X)^3-(a-\mu_X)^3}{3}\end{aligned} \tag{5.33}$$

Δεδομένου ότι:

$$b-\mu_X = b-\frac{b+a}{2} = \frac{b-a}{2} \tag{5.34a}$$

$$a-\mu_X = a-\frac{b+a}{2} = -\frac{b-a}{2} \tag{5.34b}$$

θα έχουμε:

$$\sigma_X^2 = \frac{1}{b-a}\frac{(b-a)^2}{12} = \frac{(b-a)^2}{12} \tag{5.35}$$

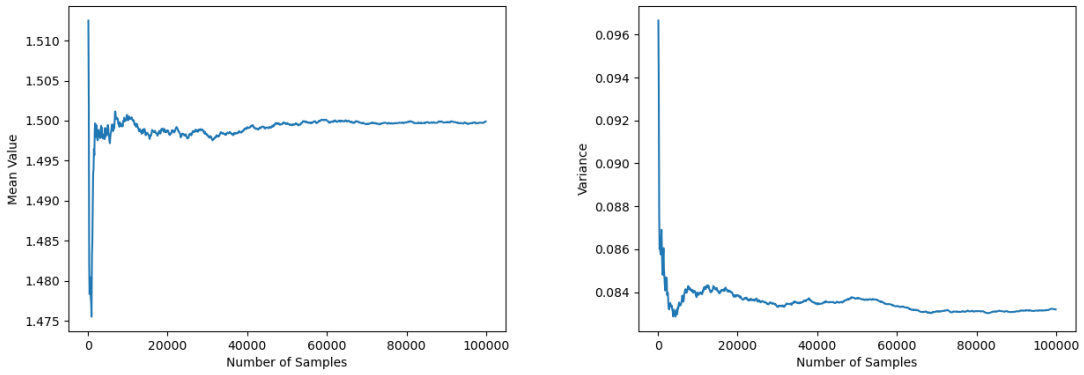
Είναι ενδιαφέρον να υπολογίσουμε την μέση τιμή και την διακύμανση και αριθμητικά. Όπως είδαμε, η αναμενόμενη τιμή $\mathbb{E}\{g(X)\}$ είναι η μέση τιμή που υπολογίζεται από τα $g(X_k)$ όπου X_k είναι δείγματα της X. Όσο περισσότερα είναι τα δείγματα, τόσο καλύτερη θα είναι η προσέγγιση για τις αναμενόμενες τιμές. Η numpy παρέχει δύο συναρτήσεις που μπορούν να χρησιμοποιηθούν για να υπολογίσουν απευθείας την μέση τιμή και την διακύμανση από τα δείγματα μίας τυχαίας μεταβλητής, την `numpy.mean` και `numpy.var` αντίστοιχα.

```
import numpy as np
import matplotlib.pyplot as plt

Nsamples = 100000
a = 1
b = 2
Nstep = 100

x = a + (b-a) * np.random.rand(Nsamples)

nsteps = np.arange(Nstep, Nsamples, Nstep, dtype = int)
varss = np.zeros( nsteps.size )
means = np.zeros( nsteps.size )

for i, n in enumerate(nsteps):
    varss[i] = np.var( x[:n] )
    means[i] = np.mean( x[:n] )

plt.close('all')
plt.figure()
plt.plot(nsteps, means ,'-')
plt.xlabel('Number of Samples')
plt.ylabel('Mean Value')
1
plt.figure()
plt.plot(nsteps, varss ,'-')
plt.xlabel('Number of Samples')
plt.ylabel('Variance')
```

Listing 5.3: meanstd.py

Στο listing 5.3 δείχνουμε πως χρησιμοποιούμε τις δύο συναρτήσεις για να υπολογίσουμε την μέση τιμή και την διακύμανση θεωρώντας διαφορετικό αριθμό n δειγμάτων κάθε φορά, ξεκινώντας από 100 δείγματα και φτάνοντας στα 100.000 δείγματα με βήμα 100 δείγματα κάθε φορά.

Στην εικόνα 5.2 δείχνουμε τα αποτελέσματα που παίρνουμε από το listing 5.2 θεωρώντας όπως και στο listing 5.2 ότι $a = 1$ και $b = 2$ οπότε σύμφωνα με την (5.32) και (5.35) περιμένουμε η μέση τιμή να είναι $\mu_X = 1.5$ και $\sigma_X^2 = 1/12$. Από την εικόνα, παρατηρούμε πως ενώ για μικρό αριθμό δειγμάτων υπάρχει σχετικά μεγάλο σφάλμα στον υπολογισμό των αναμενόμενων τιμών, εντούτοις όσο ο αριθμός των δειγμάτων αυξάνει οι τιμές που παίρνουμε από τις `numpy.mean` και `numpy.var`, πλησιάζουν πολύ καλά τις θεωρητικές τους τιμές.



(a) (b)

Εικόνα 5.2: Τα αποτελέσματα της (a) `numpy.mean` και (b) `numpy.var` για διαφορετικό αριθμό δειγμάτων

5.2.3 Κανονική Τυχαία Μεταβλητή

Αν και η ομοιόμορφη τυχαία μεταβλητή είναι αρκετά απλή, ωστόσο ο θόρυβος στα συστήματα επικοινωνιών ακολουθεί συνήθως μία διαφορετική κατανομή την οποία ονομάζουμε *κανονική κατανομή* και οι αντίστοιχες μεταβλητές ονομάζονται *κανονικές τυχαίες μεταβλητές*. Η πυκνότητα πιθανότητας μίας κανονικής τυχαίας μεταβλητής X δίνεται από την σχέση:

$$f_X(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{5.36}$$

Οι παράμετροι μ και σ σχετίζονται με αναμενόμενες τιμές της X. Αποδεικνύεται ότι η μέση τιμή και η διακύμανση της κανονικής κατανομής είναι ίσες με μ και σ^2 αντίστοιχα, δηλαδή:

$$\mu_X = \mu \tag{5.37a}$$

$$\sigma_X^2 = \sigma^2 \tag{5.37b}$$

Ωστόσο τα πράγματα είναι λίγο πιο σύνθετα αν προσπαθήσουμε να υπολογίσουμε την CDF της κανονικής κατανομής η οποία δίνεται από την σχέση:

$$F_X(y) = \int_{-\infty}^{y} f_X(x)\mathrm{d}x = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{y} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \mathrm{d}x \tag{5.38}$$

Αν κάνουμε την αλλαγή μεταβλητής $z = (x-\mu)/\sigma$ οπότε και $\mathrm{d}x = \sigma\mathrm{d}z$, τότε το ολοκλήρωμα μπορεί να γραφτεί:

$$F_X(y) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z_1} \exp\left(-\frac{z^2}{2}\right) \mathrm{d}z \tag{5.39}$$

όπου το άνω όριο ολοκλήρωσης έχει μετασχηματιστεί σε $z_1 = (y-\mu)/\sigma$. Μπορούμε να γράψουμε λίγο διαφορετικά την παραπάνω συνάρτηση. Αν χρησιμοποιήσουμε το γεγονός ότι:

$$\frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} \exp\left(-\frac{z^2}{2}\right) \mathrm{d}z = 1 \tag{5.40}$$

θα εχουμε:

$$F_X(y) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} \exp\left(-\frac{z^2}{2}\right) \mathrm{d}z + \frac{1}{\sqrt{2\pi}} \int_{+\infty}^{z_1} \exp\left(-\frac{z^2}{2}\right) \mathrm{d}z = 1 - \frac{1}{\sqrt{2\pi}} \int_{z_1}^{+\infty} \exp\left(-\frac{z^2}{2}\right) \mathrm{d}z \tag{5.41}$$

Στην βιβλιογραφία η συνάρτηση:

$$Q(y) = \frac{1}{\sqrt{2\pi}} \int_{y}^{+\infty} \exp\left(-\frac{z^2}{2}\right) \mathrm{d}z \tag{5.42}$$

αναφέρεται ως συνάρτηση Q και θα δούμε ότι παίζει πολύ σημαντικό ρόλο στην ανάλυση συστημάτων επικοινωνιών. Η CDF δίνεται από την σχέση:

$$F_X(y) = 1 - Q(y) \tag{5.43}$$

Το ολοκλήρωμα στην (5.42) δεν μπορεί να υπολογιστεί βάσει κάποιας μαθηματικής σχέσης αλλά υπάρχουν αριθμητικές μέθοδοι για αυτό τον σκοπό. Η Python έχει μία βιβλιοθήκη, την `scipy` η οποία παρέχει την `scipy.special.erfc` που υπολογίζει την συνάρτηση $\mathrm{erfc}(y)$ η οποία δίνεται από μία παρόμοια σχέση:

$$\mathrm{erfc}(y) = \frac{2}{\sqrt{\pi}} \int_{y}^{+\infty} \exp\left(-t^2\right) \mathrm{d}t \tag{5.44}$$

Αν στην (5.42) θέσουμε $z = t\sqrt{2}$ τότε έχουμε:

$$Q(y) = \frac{1}{\sqrt{2\pi}} \int_{y/\sqrt{2}}^{+\infty} \exp\left(-t^2\right) \sqrt{2}\mathrm{d}t = \frac{1}{\sqrt{\pi}} \int_{y/\sqrt{2}}^{+\infty} \exp\left(-t^2\right) \mathrm{d}t = \frac{1}{2}\mathrm{erfc}\left(\frac{y}{\sqrt{2}}\right) \tag{5.45}$$

Η παραπάνω σχέση συνδέει την συνάρτηση $Q(y)$ με την συνάρτηση $\mathrm{erfc}(y)$ και θα την χρησιμοποιήσουμε για να υπολογίσουμε την CDF της κανονικής κατανομής. Αρχικά κάνουμε δύο μικρές προσθήκες που φαίνονται στο lising 5.4 στην `commlib` για να υπολογίζουμε την συνάρτηση Q βάσει της (5.45) και την CDF βάσει της (5.43).

```
255 def Q(x):
256     return 0.5 * erfc( x / np.sqrt(2) )
257
258 def normcdf(x, mu, sigma):
259     z = (x - mu) / sigma
260     return 1 - Q(z)
```

Listing 5.4: Προσθήκες στην `commlib`

```python
import numpy as np
import matplotlib.pyplot as plt
from commlib import calc_pdfcdf, normcdf

Nsamples = 100000
mu = 2
sigma = 1
Nplot = 1000


x = np.random.randn(Nsamples) * sigma + mu
plt.close('all')
plt.plot(x[:Nplot],'s')
plt.xlabel('Sample Index')
plt.ylabel('Sample Value')

y = np.linspace(mu-3*sigma, mu+3*sigma, 100)
pdf, cdf = calc_pdfcdf(x, y)

pdf2 = 1 / np.sqrt(2 * np.pi) * np.exp( - (y-mu) ** 2.0 / 2 / sigma ** 2.0)
plt.figure()
plt.plot(y, pdf, 's', label = 'numerical', markerfacecolor="None")
plt.plot(y, pdf2, '--', label = 'analytical')
plt.xlabel('y')
plt.ylabel('PDF')
plt.title('PDF of Normal Samples')
plt.legend()

cdf2 = normcdf(y, mu, sigma)
plt.figure()
plt.plot(y, cdf, 's', label = 'numerical', markerfacecolor="None", markevery =
    3)
plt.plot(y, cdf2, '--', label = 'analytical')
plt.xlabel('y')
plt.ylabel('CDF')
plt.title('CDF of Normal Samples')
plt.legend()
```

Listing 5.5: `normal.py`

Η βιβλιοθήκη `numpy` παρέχει την συνάρτηση `numpy.random.randn` που παράγει τυχαία δείγματα μίας κανονικής κατανομής. Στο listing 5.5 χρησιμοποιούμε την `randn` για να παράγουμε δείγματα μίας κανονικής τυχαίας μεταβλητής Z. Τα δείγματα που παράγονται από την `randn` έχουν μέση τιμή μηδέν και διακύμανση ίση με 1, δηλαδή

$$\mu_Z = \mathbb{E}\{Z\} = 0 \tag{5.46a}$$

$$\sigma_Z^2 = \mathbb{E}\left\{(Z-\mu_Z)^2\right\} = 1 \tag{5.46b}$$

και επομένως η PDF της Z είναι

$$f_Z(z) = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{z^2}{2}\right) \tag{5.47}$$

Στο listing 5.5, υπάρχει μία γραμμή που μετασχηματίζει τα δείγματα της `randn` ως εξής:

```
x = np.random.randn(Nsamples) * sigma + mu
```

που είναι μαθηματικά ισοδύναμο με:

$$X = \sigma Z + \mu \tag{5.48}$$

Ας θεωρήσουμε τώρα την πυκνότητα πιθανότητας της X που προσεγγίζεται με την (5.20). Έχουμε:

$$\begin{aligned} f_X(x) &\approx \frac{1}{\Delta x}\Pr\left\{x-\tfrac{1}{2}\Delta x \le X \le x+\tfrac{1}{2}\Delta x\right\} = \\ &\frac{1}{\Delta x}\Pr\left\{x-\tfrac{1}{2}\Delta x-\mu \le X-\mu \le x+\tfrac{1}{2}\Delta x-\mu\right\} = \\ &\frac{1}{\sigma}\frac{1}{\Delta x/\sigma}\Pr\left\{\frac{x-\mu}{\sigma}-\frac{\Delta x}{2\sigma} \le \frac{X-\mu}{\sigma} \le \frac{x-\mu}{\sigma}+\frac{\Delta x}{2\sigma}\right\} \end{aligned} \tag{5.49}$$

Η πυκνότητα πιθανότητας της Z προσεγγίζεται από μία παρόμοια σχέση:

$$f_Z(z) \approx \frac{1}{\Delta z}\Pr\left\{z-\tfrac{1}{2}\Delta z \le Z \le z+\tfrac{1}{2}\Delta z\right\} \tag{5.50}$$

Συγκρίνοντας την (5.49) και της (5.50) μας υποδεικνύει ότι εφόσον $Z = (X-\mu)/\sigma$, αν θέσουμε $\Delta z = \Delta x/\sigma$ θα έχουμε:

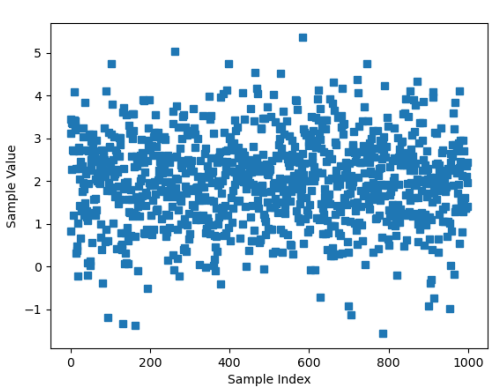
$$f_X(x) \approx \frac{1}{\sigma} f_Z\left(\frac{x-\mu}{\sigma}\right) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{5.51}$$

Επομένως με αυτόν τον τρόπο δείχνουμε ότι μετασχηματισμένη μεταβλητή ακολουθεί την επιθυμητή κατανομή. Στην εικόνα 5.3a δείχνουμε τα 1000 πρώτα δείγματα που παράγονται από τον μετασχηματισμό της `randn`. Σε σύγκριση με τα δείγματα της `rand` στην εικόνα 5.1a, βλέπουμε ότι τα δείγματα της `randn` παρουσιάζουν την τάση να συγκεντρώνονται πιο κοντά στην μέση τιμή της κατανομής $\mu_X = 2$ παρά μακριά από αυτήν. Για το λόγο αυτό υπάρχουν σχετικά λίγα δείγματα που είναι μεγαλύτερα από 4 ή μικρότερα από 0. Στην εικόνα 5.3b δείχνουμε την PDF που υπολογίζεται αριθμητικά από την `calc_pdfcdf` και θεωρητικά από την (5.36). Παρατηρούμε ότι η θεωρητική και η αριθμητική PDF συμφωνούν πολύ καλά. Στην εικόνα 5.3c δείχνουμε την CDF που υπολογίζεται αριθμητικά από τα δείγματα και την θεωρητική της μορφή που καθορίζεται από την (5.43). Παρατηρούμε ότι η συμφωνία είναι εξαιρετική.

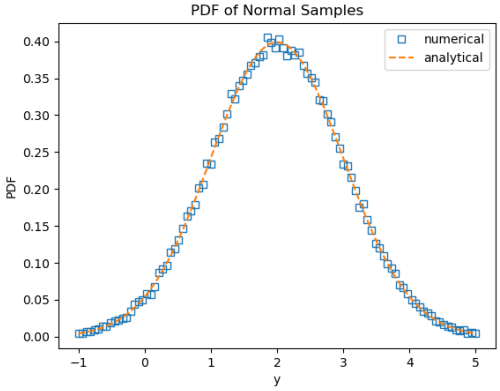
Υπάρχει μία χρήσιμη ιδιότητα του αθροίσματος δύο τυχαίων κανονικών μεταβλητών. Αν X, Y δύο τυχαίες κανονικές μεταβλητές με μέσες τιμές μ_X και μ_Y αντίστοιχα και διακυμάνσεις σ_X^2 και σ_Y^2 αντίστοιχα τότε το άθροισμα τους $Z = X + Y$ είναι και αυτό κανονική κατανομή με μέση τιμή μ_Z και διακύμανση σ_Z^2 που δίνονται αντίστοιχα από τις σχέσεις:

$$\mu_Z = \mu_X + \mu_Y \tag{5.52a}$$

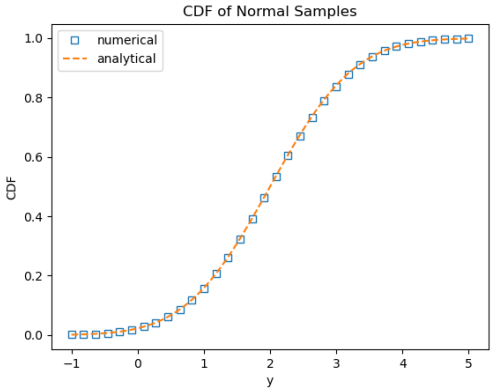
$$\sigma_Z^2 = \sigma_X^2 + \sigma_Y^2 \tag{5.52b}$$



(a)



(b)



(c)

Εικόνα 5.3: (a) Δείγματα που παράγονται από την `np.random.randn`, (b) και (c) η PDF και η CDF αντίστοιχα που υπολογίζονται αριθμητικά από την `calc_pdfcdf`, μαζί με την αναλυτική τους τιμή.

5.3 Συσχέτιση τυχαίων μεταβλητών

Στα επόμενα κεφάλαια θα αναφερόμαστε συχνά στην συσχέτιση μεταξύ δύο ή περισσότερων τυχαίων μεταβλητών. Ας υποθέσουμε ότι έχουμε δύο ζάρια τα οποία τα ρίχνουμε και καταγράφουμε το αποτέλεσμα που έρχεται. Αν το πρώτο ζάρι αντιστοιχεί στην τυχαία μεταβλητή X και το δεύτερο στην τυχαία μεταβλητή Y τότε μπορούμε να υπολογίσουμε την πιθανότητα το πρώτο ζάρι να έρθει i *και* το δεύτερο να έρθει k με $1 \leq i,k \leq 6$,

$$\Pr\{X = i, Y = k\} \tag{5.53}$$

Ας θεωρήσουμε τώρα το πηλίκο: ¨

$$\frac{\Pr\{X = i, Y = k\}}{\Pr\{Y = k\}} \tag{5.54}$$

Το πηλίκο στην (5.54) ονομάζεται *κατά συνθήκη* πιθανότητα $\Pr\{X = i|Y = k\}$ του να συμβεί το $X = i$, δεδομένου ότι έχει συμβεί το $Y = k$,

$$\Pr\{X = i|Y = k\} = \frac{\Pr\{X = i, Y = k\}}{\Pr\{Y = k\}} \tag{5.55}$$

Έχει μία αξία να ερμηνεύσουμε την κατά συνθήκη πιθανότητα στο (5.55). Αν ρίξουμε N φορές τα δύο ζάρια και μετρήσουμε πόσες φορές N_k έχει έρθει το δεύτερο ζάρι $Y = k$, αυτές θα είναι $N_k = N\Pr\{Y = k\}$. Το πλήθος των φορών N_{ik} που έχει έρθει το πρώτο $X = i$ και το δεύτερο $Y = k$ είναι $N\Pr\{X = i, Y = k\}$. Το πηλίκο N_{ik}/N_i είναι η πιθανότητα να έρθει το $X = i$ δεδομένου ότι έχει έρθει $Y = k$. Στην περίπτωση των ζαριών είναι φυσικό να θεωρήσουμε ότι το ένα ζάρι δεν επηρεάζει το άλλο. Επομένως η πιθανότητα να έρθει $X = i$ δεν εξαρτάται από το τι έχει έρθει το Y και επομένως για κάθε k θα πρέπει να έχουμε απλά:

$$\Pr\{X = i|Y = k\} = \Pr\{X = i\} \tag{5.56}$$

Καταλήγουμε επομένως στο ίδιο αποτέλεσμα που θα καταλήγαμε αν δεν λαμβάναμε υπόψη το δεύτερο ζάρι. Το δεύτερο ζάρι *δεν επηρεάζει καθόλου* το πρώτο και επομένως λέμε πως οι τυχαίες μεταβλητές X και Y είναι *ανεξάρτητες*. Προσέξτε ότι για να ισχύει αυτό θα πρέπει να ισχύει η (5.56), ή ισοδύναμα:

$$\Pr\{X = i, Y = k\} = \Pr\{X = i\}\Pr\{Y = k\} \tag{5.57}$$

που είναι η απαραίτητη συνθήκη για να πούμε ότι οι X και οι Y είναι ανεξάρτητες. Αν δεν ισχύει η (5.57), τότε οι X και Y λέμε ότι είναι *εξαρτημένες*. Για τις συνεχείς τυχαίες διαδικασίες τα πράγματα είναι παρόμοια, μόνο που αντί για τις πιθανότητες θα πρέπει να θεωρήσουμε τις πυκνότητες πιθανότητας. Σε αναλογία με την (5.23), ορίζουμε την συνδυασμένη πυκνότητα πιθανότητας $f_{XY}(x,y)$ ως εξής:

$$f_{XY}(x,y) = \lim_{\Delta x,\Delta y \to 0} \frac{\Pr\left\{x - \frac{1}{2}\Delta x \leq X \leq x + \frac{1}{2}\Delta x, y - \frac{1}{2}\Delta y \leq Y \leq y + \frac{1}{2}\Delta y\right\}}{\Delta x \Delta y} \tag{5.58}$$

Η συνδυασμένη πυκνότητα πιθανότητας χρησιμοποιείται για τον υπολογισμό αναμενόμενων τιμών της μορφής $\mathbb{E}\{g(X,Y)\}$, ως εξής:

$$\mathbb{E}\{g(X,Y)\} = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} g(x,y) f_{XY}(x,y)\mathrm{d}x\mathrm{d}y \tag{5.59}$$

Για να είναι ανεξάρτητες δύο συνεχείς τυχαίες μεταβλητές X και Y θα πρέπει να ισχύει:

$$f_{XY}(x,y) = f_X(x)f_Y(y) \tag{5.60}$$

όπου $f_X(x)$ και $f_Y(y)$ είναι οι πυκνότητες πιθανότητας των X και Y, αντίστοιχα. Για τις αναμενόμενες τιμές της μορφής $\mathbb{E}\{g(X)h(Y)\}$ όπου τα X και Y είναι ανεξάρτητες τυχαίες μεταβλητές μπορούμε να γράψουμε:

$$\mathbb{E}\{g(X)h(Y)\} = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} g(x)h(y)f_{XY}(x,y)\mathrm{d}x\mathrm{d}y = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} g(x)h(y)f_X(x)f_Y(y)\mathrm{d}x\mathrm{d}y = \int_{-\infty}^{+\infty} g(x)f_X(x)\int_{-\infty}^{+\infty} h(y)f_Y(y)\mathrm{d}x\mathrm{d}y = \left(\int_{-\infty}^{+\infty} g(x)f_X(x)\mathrm{d}x\right)\left(\int_{-\infty}^{+\infty} h(y)f_Y(y)\mathrm{d}y\right) = \mathbb{E}\{g(X)\}\mathbb{E}\{h(Y)\} \tag{5.61}$$

Επομένως η αναμενόμενη τιμή $\mathbb{E}\{g(X)h(Y)\}$ ενός γινομένου συναρτήσεων $g(X)h(Y)$ ανεξαρτήτων μεταβλητών X και Y ισούται με το γινόμενο των αναμενόμενων τιμών $\mathbb{E}\{g(X)\}\mathbb{E}\{h(Y)\}$.

Στο listing 5.6, δείχνουμε ένα παράδειγμα τέτοιων υπολογισμών. Θεωρούμε δύο τυχαίες μεταβλητές X και Y που έχουν ομοιόμορφη κατανομή στο $[0,1]$. Εφόσον τα περιμένουμε οι αντίστοιχες μεταβλητές X και Y να είναι ανεξάρτητες. Στη συνέχεια υπολογίζουμε δύο νέες τυχαίες μεταβλητές Z και W ως εξής:

$$Z = X + 2Y \tag{5.62a}$$

$$W = 2X + Y \tag{5.62b}$$

Ορίζουμε τις συναρτήσεις $g(x)$ και $h(y)$ ως εξής στο παράδειγμα μας:

$$g(x) = x^2 \tag{5.63a}$$

$$h(y) = \sqrt{y} \tag{5.63b}$$

Στο listing 5.6 υπολογίζουμε αριθμητικά την αναμενόμενη τιμή $\mathbb{E}\{g(X)h(Y)\}$ και την συγκρίνουμε με το γινόμενο $\mathbb{E}\{g(X)\}\mathbb{E}\{h(Y)\}$. Βρίσκουμε $\mathbb{E}\{g(X)h(Y)\} \approx \mathbb{E}\{g(X)\}\mathbb{E}\{h(Y)\} \approx 0.220$ οπότε στην περίπτωση των τυχαίων μεταβλητών X και Y που είναι ανεξάρτητες ισχύει η (5.61). Στην περίπτωση των μεταβλητών W και Z όμως, οι αριθμητικές τιμές που υπολογίζονται από το listing 5.6 λαμβάνουμε $\mathbb{E}\{g(Z)h(W)\} \approx 3.565$ και $\mathbb{E}\{g(Z)\}\mathbb{E}\{h(W)\} \approx 3.144$ οπότε είναι φανερό ότι $\mathbb{E}\{g(Z)h(W)\} \neq \mathbb{E}\{g(Z)\}\mathbb{E}\{h(W)\}$ επειδή οι Z και W είναι εξαρτημένες.

```
import numpy as np

Ni = 10000
X = np.random.rand(Ni)
Y = np.random.rand(Ni)

Z = X + 2 * Y
```

```
W = 2 * X + Y

g = lambda x : x ** 2.0
h = lambda y : np.sqrt(y)

EgX = np.mean( g(X) )
EhY = np.mean( h(Y) )
EgXhY = np.mean( g(X) * h(Y) )

print('Expected Value of g(X)h(Y) : ', EgXhY)
print('Expected Value of g(X) times expected value of h(Y) : ', EgX * EhY)

EgZ = np.mean( g(Z) )
EhW = np.mean( h(W) )
EgZhW = np.mean( g(Z) * h(W) )

print('Expected Value of g(Z)h(W) : ', EgZhW)
print('Expected Value of g(Z) times expected value of h(W) : ', EgZ * EhW)
```

Listing 5.6: expectedvalues.py

5.3.1 Συσχετισμένες Κανονικές Τυχαίες Μεταβλητές

Στην περίπτωση όπου οι X και η Y είναι συσχετισμένες τυχαίες μεταβλητές τότε η συνδυασμένη πυκνότητα πιθανότητας τους θα δίνεται από την σχέση:

$$f_{XY}(x,y) = \frac{1}{2\pi\sigma_X\sigma_Y\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}\left[\frac{(x-\mu_X)^2}{\sigma_X^2} + \frac{(y-\mu_Y)^2}{\sigma_Y^2} - 2\rho\frac{(x-\mu_X)(Y-\mu_Y)}{\sigma_X\sigma_Y}\right]\right) \tag{5.64}$$

όπου τα μ_X, μ_Y είναι οι μέσες τιμές της X και Y ενώ σ_X^2 και σ_Y^2 είναι οι διακυμάνσεις τους και όπως είδαμε στα προηγούμενα καθορίζονται από τις παρακάτω σχέσεις:

$$\mu_X = \mathbb{E}\{X\} \tag{5.65a}$$

$$\mu_Y = \mathbb{E}\{Y\} \tag{5.65b}$$

$$\sigma_X^2 = \mathbb{E}\{(X-\mu_X)^2\} \tag{5.65c}$$

$$\sigma_Y^2 = \mathbb{E}\{(Y-\mu_Y)^2\} \tag{5.65d}$$

Η παράμετρος ρ ονομάζεται *συντελεστής συσχέτισης* των X και Y. Υπολογίζεται ως εξής:

$$\rho = \frac{\mathbb{E}\{(X-\mu_X)(Y-\mu_Y)\}}{\sigma_X\sigma_Y} \tag{5.66}$$

Ο συντελεστής συσχέτισης όπως ορίζεται στην (5.66) χρησιμοποιείται και σε περιπτώσεις όπου τα X και Y δεν είναι απαραίτητα κανονικές τυχαίες μεταβλητές.

Στην περίπτωση όπου οι μεταβλητές είναι ανεξάρτητες, θα έχουμε:

$$\begin{aligned}\mathbb{E}\{(x-\mu_X)(Y-\mu_Y)\} &= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}(x-\mu_X)(Y-\mu_Y)f_{XY}(x,y)\mathrm{d}x\mathrm{d}y \\ &= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}(x-\mu_X)(Y-\mu_Y)f_X(x)f_Y(y)\mathrm{d}x\mathrm{d}y = \\ &= \int_{-\infty}^{+\infty}(x-\mu_X)f_X(x)\mathrm{d}x\int_{-\infty}^{+\infty}(y-\mu_Y)f_Y(y)\mathrm{d}y = 0 \quad (5.67)\end{aligned}$$

Στην παραπάνω σχέση χρησιμοποιήσαμε ότι:

$$\mu_X = \int_{-\infty}^{+\infty} x f_X(x)\mathrm{d}x \tag{5.68a}$$

$$\mu_Y = \int_{-\infty}^{+\infty} y f_Y(y)\mathrm{d}y \tag{5.68b}$$

οπότε:

$$\int_{-\infty}^{+\infty}(x-\mu_X)f_X(x)\mathrm{d}x = 0 \tag{5.69a}$$

$$\int_{-\infty}^{+\infty}(y-\mu_Y)f_Y(y)\mathrm{d}y = 0 \tag{5.69b}$$

Επομένως ο συντελεστής συσχέτισης για δύο ανεξάρτητες μεταβλητές θα είναι $\rho = 0$. Αν οι X και Y είναι κανονικές ανεξάρτητες τυχαίες μεταβλητές τότε η PDF (5.64) θα γραφτεί ως εξής:

$$f_{XY}(x,y) = \frac{1}{2\pi\sigma_X\sigma_Y}\exp\left(-\frac{(x-\mu_X)^2}{2\sigma_X^2} - \frac{(y-\mu_Y)^2}{2\sigma_Y^2}\right) \tag{5.70}$$

Οι πυκνότητες πιθανότητας των X και Y δίνονται από τις σχέσεις:

$$f_X(x) = \frac{1}{\sigma_X\sqrt{2\pi}}\exp\left(-\frac{(x-\mu_X)^2}{2\sigma_X^2}\right) \tag{5.71a}$$

$$f_Y(y) = \frac{1}{\sigma_Y\sqrt{2\pi}}\exp\left(-\frac{(y-\mu_Y)^2}{2\sigma_Y^2}\right) \tag{5.71b}$$

Συνδυάζοντας τις (5.71) και (5.70) βλέπουμε ότι το γεγονός ότι $\rho = 0$ συνεπάγεται ότι:

$$f_{XY}(x,y) = f_X(x)f_Y(y) \tag{5.72}$$

Επομένως προκύπτει ότι:

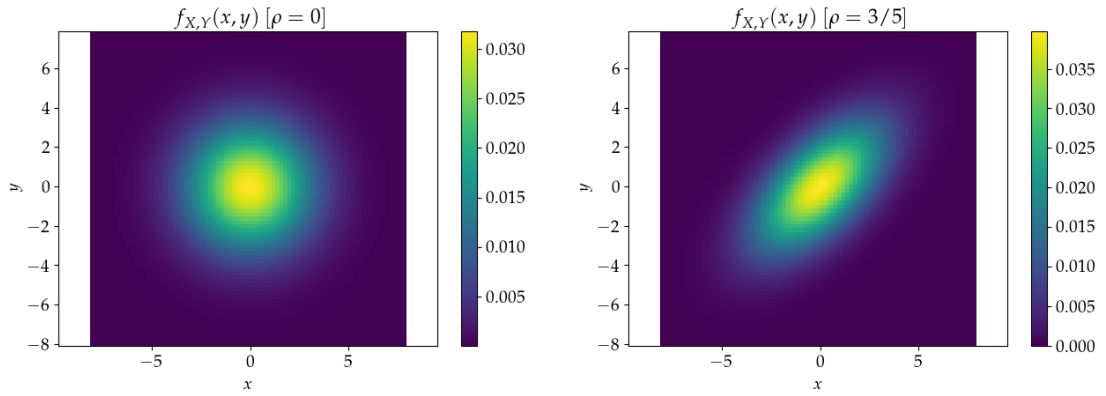
- στην περίπτωση όπου δύο μεταβλητές X και Y είναι ανεξάρτητες τότε έπεται ότι $\rho = 0$.

- ειδικά για την περίπτωση των κανονικών τυχαίων μεταβλητών, αν $\rho = 0$ τότε οι μεταβλητές X και Y είναι ανεξάρτητες.

```python
import matplotlib.pyplot as plt

import numpy as np

plt.rcParams.update({
    "text.usetex": True,
    "font.family": "serif",
    "font.serif": ["Palatino"],
    "font.size" : 14,
    "lines.linewidth" : 2,
})

def gausspdf2D(x, y, r, s2):
    expo=- 1.0/2.0/(1 - r**2) * (x**2/s2 + y**2/s2 - 2.0*r*x*y/s2)
    pdf = 1.0/2.0/np.sqrt(1 - r**2)/ np.pi /s2 * np.exp(expo)
    return pdf

def plot_2D(x, y, pdf, title):
    plt.pcolor(x,y,pdf, shading='auto')
    plt.colorbar()
    plt.axis('equal')
    plt.title(title)
    plt.xlabel('$x$')
    plt.ylabel('$y$')
    plt.tight_layout()

dx = 0.2
dy = 0.2
r = 3.0/5.0
s2 = 5

# generate 2 2d grids for the x & y bounds
y = np.arange(-8, +8, dy)
x = np.arange(-8, +8, dx)


yy, xx = np.meshgrid(y, x)
plt.close('all')

pdf1 = gausspdf2D(xx ,yy, r , s2)
plt.figure(1)
plot_2D(x, y, pdf1, r'$f_{X,Y}(x,y)$ $[\rho=3/5]$')

pdf2 = gausspdf2D(xx,yy, 0 , s2)
plt.figure(2)
plot_2D(x, y, pdf2, r'$f_{X,Y}(x,y)$ $[\rho=0]$')
```

Listing 5.7: bivariate.py

Μπορούμε να χρησιμοποιήσουμε την Python για να καταλάβουμε λίγο καλύτερα την συσχέτιση δύο τυχαίων κανονικών μεταβλητών. Ας αρχίσουμε με την PDF που δίνεται από την (5.64). Χρησιμοποιούμε την βιβλιοθήκη matplotlib για να κάνουμε γραφική παράσταση της $f_{XY}(x,y)$ στις τρεις διαστάσεις. Χρησιμοποιούμε τον κώδικα που δείχνουμε στο listing 5.7. Πριν



(a) (b)

Εικόνα 5.4: Συνδυασμένη PDF $f_{XY}(x,y)$ για δύο κανονικές μεταβλητές X και Y με $\mu_X = \mu_Y = 0$, $\sigma_X^2 = \sigma_Y^2 = 5$ και (a) $\rho = 0$ και (b) $\rho = 3/5$.

κάνουμε κάποια γραφική παράσταση χρησιμοποιούμε την `matplotlib.pyplot.rcParams.update` για να επιφέρουμε κάποιες αισθητικές αλλαγές στο πως κάνουμε γραφικές παραστάσεις - κυρίως για να χρησιμοποιούμε Latex. Στη συνέχεια η `gamespdf2D`, χρησιμοποιείται για να υπολογίζει την $f_{XY}(x,y)$ σύμφωνα και με την (5.64). Η `plot_2D`, κάνει γραφική παράσταση συναρτήσεων $z = g(x,y)$ δύο μεταβλητών x και y χρησιμοποιώντας μία κλίμακα χρωμάτων για τις τιμές του z. Στη συνέχεια φτιάχνουμε δύο άξονες, έναν για την μεταβλητή x και μία για την μεταβλητή y:

```
y = np.arange(-8, +8, dy)
x = np.arange(-8, +8, dx)
```

Χρησιμοποιώντας την `numpy.meshgrid` φτιάχνουμε δύο δισδιάστατα array, τα `xx` και `yy` που χρησιμοποιούνται για τον υπολογισμό της PDF σε δύο διαστάσεις. Οι πίνακες αυτοί έχουν στοιχεία:

```
yy[i,k] = y[k]
xx[i,k] = x[i]
```

και χρησιμοποιούνται για τον υπολογισμό της PDF. Στην εικόνα 5.4a δείχνουμε $f_{XY}(x,y)$ για $\mu_X = \mu_Y = 0$, $\sigma_X^2 = \sigma_Y^2 = 5$ και $\rho = 0$ ενώ στην εικόνα 5.4b την PDF για τις ίδιες παραμέτρους αλλά $\rho = 3/5$. Έχει ενδιαφέρον να ερμηνεύσουμε τα αποτελέσματα. Στην εικόνα 5.4a, η $f_{XY}(x,y)$ έχει μία κυκλική συμμετρία ενώ στην 5.4b έχουμε μία στραμμένη έλλειψη, η οποία οφείλεται στη συσχέτιση των X και Y ($\rho \neq 0$). Η μορφή αυτή της PDF στην 5.4b έχει ως αποτέλεσμα για παράδειγμα ότι εάν $X > 0$ τότε είναι πιο πιθανό να έχουμε $Y > 0$ (δηλαδή να είμαστε στο πρώτο τεταρτημόριο, $X > 0$ και $Y > 0$) παρά $Y < 0$ (δηλαδή να είμαστε στο τέταρτο τεταρτημόριο, $X > 0$ και $Y < 0$) επειδή η PDF στο πρώτο τεταρτημόριο είναι πιο "φωτεινή" από ότι στο τέταρτο.

5.4 Τυχαίες Διαδικασίες

Ο θόρυβος $n(t)$ στις επικοινωνίες είναι τυχαίος και μεταβάλλεται με τον χρόνο. Αυτό σημαίνει ότι για κάθε χρονική στιγμή $t = t_1, t_2, \ldots$, οι τιμές του, $n_1 = n(t_1), n_2 = n(t_2), \ldots$ είναι τυχαίες

μεταβλητές. Ανάλογα με την φύση του θορύβου τα n_i μπορεί να είναι ανεξάρτητες ή εξαρτημένες τυχαίες μεταβλητές. Τέτοιου είδους σήματα των οποίων οι τιμές σε κάθε χρονική στιγμή είναι τυχαίες μεταβλητές ονομάζονται *τυχαία σήματα* ή *τυχαίες διαδικασίες*. Ας δούμε ένα παράδειγμα. Θεωρούμε το σήμα:

$$X(t) = A\cos(2\pi f_0 t + \Phi) \tag{5.73}$$

όπου το Φ είναι μία τυχαία μεταβλητή που ακολουθεί ομοιόμορφη κατανομή με πυκνότητας πιθανότητας

$$f_\Phi(\phi) = \begin{cases} \frac{1}{2\pi} & ,0 \le \phi < 2\pi \\ 0 & ,\text{διαφορετικά} \end{cases} \tag{5.74}$$

Η τυχαία μεταβλητή Φ κάνει την $X(t)$ να είναι τυχαίο σήμα επειδή για κάθε t η τιμή του είναι τυχαία. Ωστόσο, οι τιμές του $X(t)$ για διαφορετικά t είναι συσχετισμένες. Αν $X_1 = X(t_1)$ και $X_2 = X(t_2)$ τότε, για την μέση τιμή μ_i των X_i, θα έχουμε:

$$\mu_i = \mathbb{E}\{X_i\} = \mathbb{E}\{X(t_i)\} = \mathbb{E}\{A\cos(2\pi f_0 t_i + \Phi)\} = \int_{-\infty}^{+\infty} A\cos(2\pi f_0 t_i + \phi) f_\Phi(\phi)\mathrm{d}\phi = \frac{A}{2\pi}\int_0^{2\pi} \cos(2\pi f_0 t_i + \phi)\mathrm{d}\phi = 0 \tag{5.75}$$

Ο συντελεστής συσχέτισης ρ των X_1 και X_2 θα είναι:

$$\rho = \mathbb{E}\{X_1 X_2\} = \mathbb{E}\left\{A^2\cos(2\pi f_0 t_1 + \Phi)\cos(2\pi f_0 t_2 + \Phi)\right\} \tag{5.76}$$

Χρησιμοποιώντας την ταυτότητα:

$$\cos a \cos b = \tfrac{1}{2}\cos(a-b) + \tfrac{1}{2}\cos(a+b) \tag{5.77}$$

γράφουμε το ρ ως εξής:

$$\rho = \frac{A^2}{2}\mathbb{E}\{\cos(2\pi f_0(t_1 - t_2))\} + \frac{A^2}{2}\mathbb{E}\{\cos(2\pi f_0(t_1 + t_2) + 2\Phi)\} \tag{5.78}$$

Η πρώτη αναμενόμενη τιμή δεν εξαρτάται από το Φ οπότε:

$$\mathbb{E}\{\cos(2\pi f_0(t_1 - t_2))\} = \cos(2\pi f_0(t_1 - t_2)) \tag{5.79}$$

Η δεύτερη αναμενόμενη τιμή υπολογίζεται περίπου όπως και πριν:

$$\mathbb{E}\{\cos(2\pi f_0(t_1 + t_2) + 2\Phi)\} = \frac{1}{2\pi}\int_0^{2\pi} \cos(2\pi f_0(t_1 + t_2) + 2\phi)\mathrm{d}\phi = 0 \tag{5.80}$$

Οπότε το ρ γράφεται ως εξής:

$$\rho = \frac{A^2}{2}\cos(2\pi f_0(t_1 - t_2)) \tag{5.81}$$

Παρατηρούμε ότι εκτός από συγκεκριμένους συνδυασμούς των t_1 και t_2 όπου $t_2 - t_1 = \frac{1}{2}(2m + 1)/f_0$ που μηδενίζουν το συνημίτονο, γενικά θα έχουμε $\rho \neq 0$ οπότε τα δείγματα $X_1 = X(t_1)$ και $X_2 = X(t_2)$ σχετίζονται δεδομένου ότι στο όρισμα του συνημιτόνου έχουν τυχαία μεν αλλά την ίδια δε τιμή για το Φ.

5.4.1 Ισχύς και Ενέργεια

Στο κεφάλαιο 3 είχαμε ορίσει την στιγμιαία ισχύ P και την ενέργεια ενός μη τυχαίου σήματος $x(t)$ μέσα στην χρονική διάρκεια $[t_a, t_b]$, ως εξής:

$$P(t) = x^2(t) \tag{5.82}$$

$$E(t_a, t_b) = \int_{t_a}^{t_b} x^2(t)\mathrm{d}t \tag{5.83}$$

Για ένα τυχαίο σήμα $X(t)$, θα μπορούσαμε να χρησιμοποιήσουμε τους ίδιους ορισμούς και να θεωρήσουμε το $X^2(t)$ και το $E_X(t_a, t_b) = \int_{t_a}^{t_b} X^2(t)\mathrm{d}t$, τα οποία όμως είναι τυχαίες μεταβλητές και επομένως ίσως είναι καλύτερο να υπολογίσουμε την αναμενόμενη ισχύ και ενέργεια ως εξής:

$$P(t) = \mathbb{E}\left\{X^2(t)\right\} \tag{5.84}$$

$$\mathscr{E}_X(t_a, t_b) = \mathbb{E}\left\{E_X(t_a, t_b)\right\} = \mathbb{E}\left\{\int_{t_a}^{t_b} X^2(t)\mathrm{d}t\right\} \tag{5.85}$$

Η μέση ισχύς στο διάστημα $[t_a, t_b]$ θα είναι απλά:

$$\mathscr{P}_X(t_a, t_b) = \frac{\mathscr{E}_X(t_a, t_b)}{t_b - t_a} \tag{5.86}$$

Συνήθως για τον υπολογισμό της μέσης ισχύος θέτουμε $t_a \to -\infty$ και $t_b \to +\infty$ και να χρησιμοποιήσουμε τον ορισμό:

$$\mathscr{P}_X = \lim_{\tau \to +\infty} \frac{\mathscr{E}_X\left(-\frac{\tau}{2}, \frac{\tau}{2}\right)}{\tau} \tag{5.87}$$

```python
import numpy as np
import matplotlib.pyplot as plt
from commlib import carrier

plt.rcParams.update({
    "text.usetex": True,
    "font.family": "serif",
    "font.serif": ["Palatino"],
    "font.size" : 14,
    "lines.linewidth" : 2,
})

f0 = 1e9
T = 1/f0
ta = 0
tb = T
Nt = 1000
Ni = 100
Nplot = 20

```

```
t = np.linspace(ta, tb, Nt)
Xts = []
EX = np.zeros( Ni )
PX = np.zeros( Ni )

for ni in range(Ni):
    phi = 2 * np.pi * np.random.rand()
    Xt = carrier(t, f0, phi = phi)
    Xts.append( Xt )
    EX[ni] = Xt.energy()
    PX[ni] = EX[ni] / (tb - ta)

plt.close('all')
plt.figure()

for n in range(Nplot):
    Xts[n].plot(line_type = '-')

plt.ylabel('$X(t)$')
plt.xlabel('$t$')
plt.tight_layout()

EXmean = np.mean(EX)
PXmean = np.mean(PX)

print('Mean Energy:', EXmean)
print('Mean Power:', PXmean)
```

Listing 5.8: randomcarrier.py

Μπορούμε να υπολογίσουμε αριθμητικά την ενέργεια και την ισχύ του τυχαίου σήματος (5.73). Χρησιμοποιούμε το listing 5.8 θεωρώντας $A = 1$, για να γεννήσουμε N διαφορετικές πραγματοποιήσεις $X_i(t)$ του $X(t)$ μέσω της κλάσης `carrier` της `commlib`. Η μέση ενέργεια $\mathscr{E}_X$ υπολογίζεται από την μέση τιμή των ενεργειών $\int_{t_a}^{t_b} X_i^2(t)\mathrm{d}t$ όλων των σημάτων $X_i(t)$ οι οποίες προκύπτουν από την μέθοδο `energy` της κλάσης `carrier`.

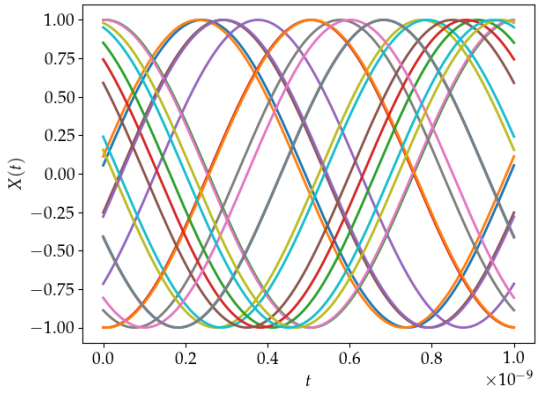
$$\mathscr{E}_X(t_a,t_b) \approx \frac{1}{N}\sum_{i=1}^{N}\int_{t_a}^{t_b} X_i^2(t)\mathrm{d}t \tag{5.88}$$

Στο listing 5.8 κάνουμε plot και τις πρώτες 20 πραγματοποιήσεις $X_i(t)$ που φαίνονται στην εικόνα 5.5 για συχνότητα φέροντος $f_0 = 1\mathrm{GHz}$. Καταλήγουμε σε ένα αποτέλεσμα $\approx 5\times 10^{-10}$ για την μέση ενέργεια του σήματος όταν θεωρούμε μία πλήρη περίοδο του σήματος $T = 1/f_0$, από $t_a = 0$ μέχρι $t_b = T$. Η μέση ισχύς στο ίδιο διάστημα υπολογίζεται σε ≈ 0.5.

Ας υπολογίσουμε και μαθηματικά την μέση ενέργεια και την μέση ισχύ του σήματος για να δούμε πως συγκρίνονται με τα αριθμητικά αποτελέσματα. Έχουμε:

$$\mathscr{E}(t_a,t_b) = \mathbb{E}\left\{\int_{t_a}^{t_b} X^2(t)\mathrm{d}t\right\} = \mathbb{E}\left\{\int_0^T A^2\cos^2(2\pi f_0 t+\Phi)\mathrm{d}t\right\} = A^2\,\mathbb{E}\left\{\int_0^T \frac{\cos(4\pi f_0 t+2\Phi)+1}{2}\mathrm{d}t\right\} = \frac{A^2T}{2} \tag{5.89}$$

Στην παραπάνω χρησιμοποιήσαμε την σχέση $\cos^2 a = \frac{1}{2}[1+\cos(2a)]$. Επίσης, εφόσον το $\cos(4\pi f_0 t+2\Phi)$ έχει περίοδο $1/(2f_0) = T/2$, το ολοκλήρωμα του μέσ σε δύο περιόδους στο



Εικόνα 5.5: Υλοποιήσεις $X_i(t)$ του τυχαίου σήματος $X(t)$ στην (5.73).

διάστημα $[0,T]$ θα είναι ίσο με μηδέν,

$$\int_0^T \cos\left(4\pi f_0 t + 2\Phi\right) \mathrm{d}t = 0 \tag{5.90}$$

οπότε προκύπτει η (5.89). Η μέση ισχύς είναι:

$$\mathscr{P}_X(t_a,t_b) = \frac{\mathscr{E}_X(t_a,t_b)}{t_b - t_a} = \frac{A^2}{2} \tag{5.91}$$

Αντικαθιστώντας βρίσκουμε ότι οι θεωρητικές τιμές των $\mathscr{P}_X(t_a,t_b)$ και $\mathscr{E}_X(t_a,t_b)$ είναι 0.5 και 5×10^{-10} αντίστοιχα, δηλαδή πολύ κοντά στις αριθμητικές τους τιμές.

Ας θεωρήσουμε και ένα δεύτερο παράδειγμα όπου έχουμε τετραγωνικούς παλμούς $p(t)$ οι οποίοι πολλαπλασιάζονται με τυχαίο πλάτος A. Το τυχαίο σήμα μπορεί να γραφτεί:

$$X(t) = Ap(t) \tag{5.92}$$

όπου το $p(t)$ είναι ένας τετραγωνικός παλμός με πλάτος 1

$$p(t) = \begin{cases} 1 & ,-T_1/2 \le t \le T_1/2 \\ 0 & ,\text{διαφορετικά} \end{cases} \tag{5.93}$$

και θεωρούμε ότι το A είναι διακριτή τυχαία μεταβλητή που παίρνει τέσσερις διαφορετικές τιμές τα ±1 και ±3 με την ίδια πιθανότητα, δηλαδή:

$$\Pr\{A=1\} = \Pr\{A=-1\} = \Pr\{A=3\} = \Pr\{A=-3\} = \frac{1}{4} \tag{5.94}$$

```
import numpy as np
import matplotlib.pyplot as plt
from commlib import square_pulse

plt.rcParams.update({
    "text.usetex": True,
```

```
    "font.family": "serif",
    "font.serif": ["Palatino"],
    "font.size" : 14,
    "lines.linewidth" : 2,
})

Tp = 1e-9
Np = 100
Ni = 10000
Nplot = 10
amps = np.array([-3, -1, 1, 3])
t = np.linspace(-0.5 * Tp, 0.5 * Tp, Np)
Xts = []
EX = np.zeros( Ni )
PX = np.zeros( Ni )

for ni in range(Ni):
    ampi = np.random.randint(0, 4)
    Xt = square_pulse(t, Tp, amp = amps[ampi])
    Xts.append( Xt )
    EX[ni] = Xt.energy()
    PX[ni] = EX[ni] / Tp

plt.close('all')
plt.figure()

for n in range(Nplot):
    Xts[n].plot(line_type = '-')

plt.ylabel('$X(t)$')
plt.xlabel('$t$')
plt.tight_layout()

EXmean = np.mean(EX)
PXmean = np.mean(PX)

print('Mean Energy:', EXmean)
print('Mean Power:', PXmean)
```

Listing 5.9: randomamp.py

Για να υπολογίσουμε την μέση ενέργεια και την ισχύ του $X(t)$ χρησιμοποιούμε το listing 5.9 το οποίο είναι παρόμοιο σε λογική με το listing 5.8. Χρησιμοποιούμε την `square_pulse` της `commlib` όπου έχουμε κάνει μία μικρή προσθήκη ώστε να μπορούμε να θέτουμε το πλάτος του παλμού τιμές οι οποίες να διαφέρουν από το 1 όπως φαίνεται και στο listing 5.10. Η μεταβλητή εισόδου amp, στην `__init__` καθορίζει το πλάτος του παλμού που μπορεί να διαφέρει από το 1.

Ο κώδικας στο listing 5.9 υπολογίζει τις τιμές $\approx 5 \times 10^{-9}$ και ≈ 5 για τις τιμές της μέσης ενέργειας $\mathscr{E}_X(t_a, t_b)$ και την ισχύος $\mathscr{P}_X(t_a, t_b)$ αντίστοιχα, στο διάστημα $t_a = -T_1/2$ και $t_b = T_1/2$. Θεωρητικά οι τιμές αυτές υπολογίζονται ως εξής:

$$\mathscr{E}_X(t_a, t_b) = \mathbb{E}\left\{\int_{-T_1/2}^{T_1/2} X^2(t)\mathrm{d}t\right\} = \mathbb{E}\left\{A^2 \int_{-T_1/2}^{T_1/2} p^2(t)\mathrm{d}t\right\} = \mathbb{E}\left\{A^2 T_1\right\} = T_1\,\mathbb{E}\left\{A^2\right\} \tag{5.95}$$

Η μέση ισχύς δίνεται από την σχέση:

$$\mathscr{P}_X(t_a,t_b) = \frac{\mathscr{E}_X(t_a,t_b)}{t_b - t_a} = \frac{\mathscr{E}_X(t_a,t_b)}{T_1}\,\mathbb{E}\left\{A^2\right\} \tag{5.96}$$

Για το $\mathbb{E}\left\{A^2\right\}$, σε αναλογία με το (5.26) θα έχουμε:

$$\mathbb{E}\left\{A^2\right\} = (-3)^2 \times \frac{1}{4} + (-1)^2 \times \frac{1}{4} + 1^2 \times \frac{1}{4} + 3^2 \times \frac{1}{4} = 5 \tag{5.97}$$

Οπότε από την (5.95) και (5.97) καταλήγουμε στην τιμή $\mathscr{E}_X(t_a,t_b) = 5T_1 = 5 \times 10^{-9}$ ενώ η μέση ισχύς είναι $\mathscr{P}_X(t_a,t_b) = 5$, οπότε οι αριθμητικές τιμές που υπολογίσαμε πριν συμφωνούν πολύ καλά με τις θεωρητικές τους τιμές.

```
141 class square_pulse(signal):
142     """
143     Square pulse
144     """
145     def __init__(self, t, T1, tcenter = 0.0, amp = 1.0):
146         self.tcenter = tcenter
147         samples = np.zeros(t.size)
148         i = np.where( np.abs (t - tcenter) <= 0.5 * T1)
149         samples[ i ] = amp
150         self.amp = amp
151         self.T1 = T1
152         super().__init__( t = t, samples = samples )
153 
154     def calc_FS_coeffs(self, m):
155         """
156         Override the parent function since this is known in closed form
157         """
158         fm = self.calc_FS_freqs(m)
159         self.Cm = self.amp * self.T1 / self.T * np.sinc( fm * self.T1 ) * np.exp
    ( -1j * 2* np.pi * fm * self.tcenter )
160         return self.Cm
```

Listing 5.10: Προσθήκες στην `commlib`

Συχνά ενδιαφερόμαστε για την μέση ισχύ στο διάστημα $(-\infty,+\infty)$ για την περίπτωση ενός περιοδικού τυχαίου σήματος όπως το (5.73). Θα εφαρμόσουμε ότι κάναμε στην περίπτωση του (3.29). Θεωρούμε πάλι την ακολουθία $\tau_n = 2nT$ όπου T είναι η περίοδος του σήματος. Θα έχουμε:

$$\lim_{n\to\,+\infty} \frac{1}{2nT}\int_{-nT}^{nT} X^2(t)\mathrm{d}t = \lim_{n\to\,+\infty} \frac{1}{2nT} \underbrace{\left(\int_{-nT}^{-(n-1)T} |X(t)|^2\mathrm{d}t + \cdots + \int_{(n-1)T}^{nT} |X(t)|^2\mathrm{d}t\right)}_{2n\ \text{όροι}}$$

$$= \frac{1}{T}\int_0^T |X(t)|^2\mathrm{d}t \tag{5.98}$$

οπότε και:

$$\mathscr{P}_X = \lim_{\tau\to\,+\infty} \frac{\mathscr{E}_X\left(-\frac{\tau}{2},\frac{\tau}{2}\right)}{\tau} = \lim_{n\to\,+\infty} \frac{\mathscr{E}_X\left(-\frac{\tau_n}{2},\frac{\tau_n}{2}\right)}{\tau_n} = \lim_{n\to\,+\infty} \frac{\mathscr{E}_X\left(-nT,nT\right)}{2nT} =$$

$$\mathbb{E}\left\{\lim_{n\to\,+\infty} \frac{1}{2nT}\int_{-nT}^{nT} X^2(t)\mathrm{d}t\right\} = \mathbb{E}\left\{\frac{1}{T}\int_0^T |X(t)|^2\mathrm{d}t\right\} = \mathscr{P}_X(0,T) \tag{5.99}$$

Όπως λοιπόν και στην περίπτωση των μη τυχαίων περιοδικών σημάτων που είδαμε στην (3.30), στα περιοδικά τυχαία σήματα, η μέση ισχύς στο διάστημα $(-\infty,+\infty)$, $\mathscr{P}_X$ ισούται με την μέση ισχύ στο διάστημα της βασικής περιόδου $[0,]$, $\mathscr{P}_X(0,T)$.

5.4.2 Συνάρτηση Αυτοσυσχέτισης και Στατικά Σήματα

Η συνάρτηση αυτοσυσχέτισης $R_{XX}(t_1,t_2)$ ορίζεται ως η αναμενόμενη τιμή του γινομένου των τιμών του τυχαίου σήματος $X_1 = X(t_1)$ και $X_2 = X(t_2)$,

$$R_{XX}(t_1,t_2) = \mathbb{E}\{X_1X_2\} = \mathbb{E}\{X(t_1)X(t_2)\} \tag{5.100}$$

Αν τα X_1 και X_2 είναι ανεξάρτητα τότε θα έχουμε

$$R_{XX}(t_1,t_2) = \mathbb{E}\{X(t_1)\}\mathbb{E}\{X(t_2)\} \tag{5.101}$$

οπότε αν συμβολίσουμε την μέση τιμή της X την χρονική στιγμή t ως:

$$\mu_X(t) = \mathbb{E}\{X(t)\} \tag{5.102}$$

θα έχουμε:

$$R_{XX}(t_1,t_2) = \mu_X(t_1)\mu_X(t_2) \tag{5.103}$$

Παράδειγμα 5.4.1: Συνάρτηση αυτοσυσχέτισης φέροντος σήματος

Ας υπολογίσουμε την συνάρτηση αυτοσυσχέτισης για το τυχαίο σήμα (5.73).

Λύση

Χρησιμοποιώντας την (5.77), Θα έχουμε:

$$R_{XX}(t_1,t_2) = \mathbb{E}\{X(t_1)X(t_2)\} = \mathbb{E}\left\{A^2\cos(2\pi f_0t_1+\Phi)\cos(2\pi f_0t_2+\Phi)\right\} = A^2\,\mathbb{E}\left\{\tfrac{1}{2}\cos(2\pi f_0(t_1+t_2)+2\Phi)+\tfrac{1}{2}\cos(2\pi f_0(t_1-t_2))\right\} = \frac{A^2}{2}\cos(2\pi f_0(t_1-t_2)) \tag{5.104}$$

Στην παραπάνω σχέση χρησιμοποιήσαμε το ότι η αναμενόμενη τιμή του $\cos(2\pi f_0(t_1+t_2)+2\Phi)$ είναι μηδέν:

$$\mathbb{E}\{\cos(2\pi f_0(t_1+t_2)+2\Phi)\} = \frac{1}{2\pi}\int_0^{2\pi}\cos(2\pi f_0(t_1+t_2)+2\phi)\mathrm{d}\phi = 0 \tag{5.105}$$

Παρατηρούμε ότι η $R_{XX}(t_1,t_2)$ στην περίπτωση αυτή εξαρτάται μόνο από την διαφορά των χρονικών στιγμών t_1-t_2,

$$R_{\mathrm{XX}}(t_1,t_2) = R_{\mathrm{XX}}(t_1-t_2) \tag{5.106}$$

Τυχαία σήματα που παρουσιάζουν την συμπεριφορά $R_{XX}(t_1,t_2) = R_{XX}(t_1 - t_2)$ όπως στο παράδειγμα 5.4.1, ονομάζονται *στατικά*. Η στατικότητα έχει ορισμένες ενδιαφέρουσες επιπτώσεις στην στατιστική συμπεριφορά του σήματος. Για παράδειγμα, η αναμενόμενη ισχύς ενός στατικού τυχαίου σήματος σύμφωνα με την (5.82) θα είναι:

$$P(t) = \mathbb{E}\left\{X^2(t)\right\} = \mathbb{E}\left\{X(t)X(t)\right\} = R_{XX}(t,t) = R_{XX}(t-t) = R_{XX}(0) \tag{5.107}$$

Σύμφωνα με την (5.107), για ένα στατικό σήμα $X(t)$, η αναμενόμενη ισχύς δεν εξαρτάται από τον χρόνο t. Η μέση ενέργεια σε ένα διάστημα $[t_a,t_b]$ είναι:

$$\mathscr{E}_X(t_a,t_b) = \mathbb{E}\left\{\int_{t_a}^{t_b} X^2(t)\mathrm{d}t\right\} = \int_{t_a}^{t_b} \mathbb{E}\left\{X^2(t)\right\}\mathrm{d}t = \int_{t_a}^{t_b} R_{XX}(0)\mathrm{d}t = R_{XX}(0)(t_b - t_a) \tag{5.108}$$

Η (5.108) δείχνει ότι για ένα στατικό σήμα η ενέργεια σε ένα οποιοδήποτε διάστημα $[t_a,t_b]$ εξαρτάται μόνο από τη διάρκεια $t_b - t_a$ του διαστήματος. Η μέση ισχύς δίνεται από την (5.86),

$$\mathscr{P}_X(t_a,t_b) = \frac{\mathscr{E}_X(t_a,t_b)}{t_b - t_a} = R_{XX}(0) \tag{5.109}$$

και επομένως όπως θα περίμενε και κανείς, δεν εξαρτάται από το χρονικό διάστημα $[t_a,t_b]$ που την θεωρούμε.

Παράδειγμα 5.4.2: **Συνάρτηση αυτοσυσχέτισης ενός ψηφιακού σήματος με τυχαία πλάτη**

Στην παράγραφο 3.6, είδαμε ότι ένα ψηφιακό σήμα μπορεί να αναπαρασταθεί από ένα άθροισμα της μορφής:

$$X(t) = \sum_k A_k p(t - kT_S) \tag{5.110}$$

όπου τα A_k θεωρούμε ότι είναι οι τιμές του ψηφιακού σήματος για $kT_S \leq t < (k+1)T_S$. Θα υπολογίσουμε την συνάρτηση αυτοσυσχέτισης $R_X X(t_1,t_2)$ θεωρώντας ότι τα A_k είναι ανεξάρτητες τυχαίες μεταβλητές με μέση τιμή $\mathbb{E}\{A_k\} = 0$ και έχουν την ίδια διακύμανση $\sigma_A^2 = \mathbb{E}\left\{A_k^2\right\}$. Στο παράδειγμα αυτό δεν κάνουμε κάποια υπόθεση για την εξίσωση που καθορίζει τους παλμούς $p(t)$ (π.χ. δεν θεωρούμε ότι είναι οπωσδήποτε τετραγωνικοί).

Λύση

Υπολογίζουμε την συνάρτηση αυτοσυσχέτισης ως εξής:

$$R_{XX}(t_1,t_2) = \mathbb{E}\left\{X(t_1)X(t_2)\right\} = \mathbb{E}\left\{\sum_k A_k p(t_1 - kT_S)\sum_q A_q p(t_2 - qT_S)\right\} =$$

$$\mathbb{E}\left\{\sum_{kq} A_k A_q p(t_1 - kT_S)p(t_2 - qT_S)\right\} = \sum_{kq}\mathbb{E}\left\{A_k A_q\right\} p(t_1 - kT_S)p(t_2 - qT_S) \tag{5.111}$$

Δεδομένου ότι οι μεταβλητές A_k είναι ανεξάρτητες θα έχουμε:

$$\mathbb{E}\left\{A_k A_q\right\} = 0 \quad \text{,όταν } k \neq q \tag{5.112}$$

ενώ όταν $k = q$, θα έχουμε $\mathbb{E}\left\{A_k A_q\right\} = \mathbb{E}\left\{A_k^2\right\} = \sigma_A^2$. Οπότε στο άθροισμα (5.111) κρατάμε μόνο τους όρους που έχουν $k = q$ και το άθροισμα γράφεται ως εξής:

$$R_{XX}(t_1,t_2) = \mathbb{E}\left\{X(t_1)X(t_2)\right\} = \sum_k \mathbb{E}\left\{A_k^2\right\} p(t_1 - kT_S)p(t_2 - kT_S) = \sigma_A^2 \sum_k p(t_1 - kT_S)p(t_2 - kT_S) \tag{5.113}$$

Αν θέτουμε $t_1 = t$ και $t_2 = t + t'$, τότε θα έχουμε:

$$R_{XX}(t,t+t') = \sigma_A^2 \sum_k p(t - kT_S)p(t + t' - kT_S) \tag{5.114}$$

Από την παραπάνω εξίσωση δεν είναι προφανές ότι το $R_{XX}(t,t+t')$ εξαρτάται μόνο από το t' οπότε δεν προκύπτει ότι η $X(t)$ είναι στατική.

Παράδειγμα 5.4.3: **Η συνάρτηση αυτοσυσχέτισης του παραδείγματος 5.4.2 θεωρώντας τετραγωνικούς παλμούς $p(t)$**

Θέλουμε να υπολογίσουμε την συνάρτηση αυτοσυσχέτισης $R_{XX}(t_1,t_2)$ στην περίπτωση του παραδείγματος 5.4.2 όταν οι παλμοί είναι τετραγωνικοί με

$$p(t) = \begin{cases} 1 & \text{,} 0 \leq t < T_S \\ 0 & \text{,διαφορετικά} \end{cases} \tag{5.115}$$

Λύση

Στην περίπτωση αυτή ισχύει η σχέση στην οποία καταλήξαμε στο παράδειγμα 5.4.2, δηλαδή:

$$R_{XX}(t_1,t_2) = \sigma_A^2 \sum_k p(t_1 - kT_S)p(t_2 - kT_S) \tag{5.116}$$

Ωστόσο στην περίπτωση μας μπορούμε να δούμε ότι το $p(t_1 - kT_S)$ είναι μη μηδενικό και ίσο με ένα μόνο όταν $kT_S \leq t_1 < (k+1)T_S$. Για να έχουμε $p(t - kT_S)p(t + t' - kT_S) \neq 0$, θα πρέπει και $kT_S \leq t_2 < (k+1)T_S$. Επομένως το $R_X X(t_1,t_2)$ είναι μη μηδενικό όταν το t_1 και το t_2 ανήκουν στην ίδια διάρκεια συμβόλου, δηλαδή υπάρχει ένας ακέραιος k για τον οποίο να ισχύουν ταυτόχρονα $kT_S \leq t_1, t_2 < (k+1)T_S$ οπότε τότε θα έχουμε $R_X X(t_1,t_2) = \sigma_A^2$ ενώ σε κάθε άλλη περίπτωση θα έχουμε $R_{XX}(t_1,t_2) = 0$.

5.4.3 Φασματική πυκνότητα ισχύος

Στην περίπτωση των τυχαίων σημάτων, μας ενδιαφέρει η *φασματική πυκνότητα ισχύος* (power spectral density - PSD) που καθορίζει σε ποιο μέρος του φάσματος έχει ισχυρή ή ασθενή παρουσία ένα τυχαίο σήμα $X(t)$. Η PSD ορίζεται και για μη τυχαία σήματα όπως είδαμε και στην παράγραφο 4.8. Ανάλογα ας ορίσουμε μία "παραθυρωμένη" έκδοση $X_\tau(t)$ ενός τυχαίου σήματος $X(t)$ ως εξής:

$$X_\tau(t) = \begin{cases} X(t) & ,|t| \le \tau/2 \\ 0 & ,\text{διαφορετικά} \end{cases} \tag{5.117}$$

Η μέση ισχύς στο $(-\infty, +\infty)$ δίνεται από την (5.87),

$$\mathscr{P}_X = \lim_{\tau \to +\infty} \mathbb{E}\left\{ \frac{1}{\tau} \int_{-\tau/2}^{\tau/2} |X(t)|^2 \mathrm{d}t \right\} = \mathbb{E}\left\{ \lim_{\tau \to +\infty} \frac{1}{\tau} \int_{-\infty}^{\infty} |X_\tau(t)|^2 \mathrm{d}t \right\} \tag{5.118}$$

Αν ορίσουμε το $\tilde{X}_\tau(f)$ να είναι ο μετασχηματισμός Fourier της τυχαίας μεταβλητής $X_\tau(t)$,

$$\tilde{X}_\tau(f) = \mathscr{F}\{X_\tau(t)\} = \int_{-\infty}^{+\infty} X_\tau(t) \mathrm{e}^{-j2\pi ft} \mathrm{d}t \tag{5.119}$$

τότε χρησιμοποιώντας την ταυτότητα του Parseval θα έχουμε:

$$\mathscr{P}_X = \int_{-\infty}^{\infty} \lim_{\tau \to +\infty} \frac{1}{\tau} \mathbb{E}\left\{ |\bar{X}_\tau(f)|^2 \right\} \mathrm{d}f \tag{5.120}$$

Αν ορίσουμε το $S_X(f)$ να δίνεται από την σχέση:

$$S_X(f) = \lim_{\tau \to +\infty} \frac{1}{\tau} \mathbb{E}\left\{ |\bar{X}_\tau(f)|^2 \right\} \tag{5.121}$$

τότε προκύπτει ότι:

$$\mathscr{P}_X = \int_{-\infty}^{+\infty} S_X(f) \mathrm{d}f \tag{5.122}$$

Η $S_X(f)$ είναι η PSD του $X(t)$. Μπορούμε να δείξουμε ότι συνδέεται με την συνάρτηση αυτοσυσχέτισης $R_{XX}(t_1, t_2)$. Έχουμε:

$$|\tilde{X}_\tau(f)|^2 = \tilde{X}_\tau(f)\tilde{X}_\tau^*(f) = \left(\int_{-\infty}^{+\infty} x_\tau(t) \mathrm{e}^{-j2\pi ft} \mathrm{d}t \right) \left(\int_{-\infty}^{+\infty} x_\tau(t) \mathrm{e}^{-j2\pi ft} \mathrm{d}t \right)^* =$$
$$\left(\int_{-\infty}^{+\infty} x_\tau(t_1) \mathrm{e}^{-j2\pi ft_1} \mathrm{d}t_1 \right) \left(\int_{-\infty}^{+\infty} x_\tau^*(t_2) \mathrm{e}^{j2\pi ft_2} \mathrm{d}t_2 \right) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x_\tau(t_1) x_\tau^*(t_2) \mathrm{e}^{j2\pi f(t_2 - t_1)} \mathrm{d}t_1 \mathrm{d}t_2 \tag{5.123}$$

και

$$\frac{1}{\tau} \mathbb{E}\left\{ |\tilde{X}_\tau(f)|^2 \right\} = \frac{1}{\tau} \mathbb{E}\left\{ \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x_\tau(t_1) x_\tau^*(t_2) \mathrm{e}^{j2\pi f(t_2 - t_1)} \mathrm{d}t_1 \mathrm{d}t_2 \right\} =$$
$$\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \frac{1}{\tau} \mathbb{E}\left\{ x_\tau(t_1) x_\tau^*(t_2) \right\} \mathrm{e}^{j2\pi f(t_2 - t_1)} \mathrm{d}t_1 \mathrm{d}t_2 = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \frac{1}{\tau} R'_{XX}(t_1, t_2) \mathrm{e}^{j2\pi f(t_2 - t_1)} \mathrm{d}t_1 \mathrm{d}t_2 \tag{5.124}$$

όπου με

$$R'_{XX}(t_1,t_2) = \mathbb{E}\left\{x_\tau(t_1)x^*_\tau(t_2)\right\} \tag{5.125}$$

έχουμε ορίσει την συνάρτηση αυτοσυσχέτισης του $X_\tau(t)$. Εφόσον $X_\tau(t) = 0$ για $|t| > \frac{\tau}{2}$, έπεται ότι θα έχουμε και $R'_{XX}(t_1,t_2) = 0$ για $|t_1| > \frac{\tau}{2}$ emphή $|t_2| > \frac{\tau}{2}$. Στην παραπάνω σχέση θέτουμε $t' = t_1 - t_2$,

$$\frac{1}{\tau}\mathbb{E}\left\{|\tilde{X}_\tau(f)|^2\right\} = \int_{-\infty}^{+\infty}\frac{1}{\tau}\int_{-\infty}^{+\infty} R'_{XX}(t'+t_2,t_2)\mathrm{e}^{-j2\pi ft'}\mathrm{d}t'\mathrm{d}t_2 = \int_{-\infty}^{+\infty}\left\{\frac{1}{\tau}\int_{-\infty}^{+\infty} R'_{XX}(t'+t_2,t_2)\mathrm{d}t_2\right\}\mathrm{e}^{-j2\pi ft'}\mathrm{d}t' \tag{5.126}$$

Για το εσωτερικό ολοκλήρωμα

$$\int_{-\infty}^{+\infty} R'_{XX}(t'+t_2,t_2)\mathrm{e}^{-j2\pi ft'}\mathrm{d}t_2 \tag{5.127}$$

χρησιμοποιούμε το ότι $R'_{XX}(t'+t_2,t_2) = 0$, όταν $|t_2| > \tau/2$ οπότε:

$$\int_{-\infty}^{+\infty} R'_{XX}(t'+t_2,t_2)\mathrm{e}^{-j2\pi ft'}\mathrm{d}t_2 = \int_{-\tau/2}^{+\tau/2} R'_{XX}(t'+t_2,t_2)\mathrm{e}^{-j2\pi ft'}\mathrm{d}t_2 \tag{5.128}$$

Ορίζουμε την

$$\bar{R}_{XX}(t') = \lim_{\tau\to\infty}\frac{1}{\tau}\int_{-\tau/2}^{+\tau/2} R'_{XX}(t'+t_2,t_2)\mathrm{d}t_2 \tag{5.129}$$

και θα έχουμε:

$$\bar{R}_{XX}(t') = \lim_{\tau\to\infty}\frac{1}{\tau}\int_{-\tau/2}^{+\tau/2} R'_{XX}(t'+t_2,t_2)\mathrm{d}t_2 = \lim_{\tau\to\infty}\frac{1}{\tau}\int_{-\tau/2}^{+\tau/2} R_{XX}(t'+t_2,t_2)\mathrm{d}t_2 \tag{5.130}$$

και γράφουμε:

$$S_X(f) = \lim_{\tau\to\infty}\frac{1}{\tau}\mathbb{E}\left\{|\tilde{X}_\tau(f)|^2\right\} = \int_{-\infty}^{+\infty}\bar{R}_{XX}(t')\mathrm{e}^{-j2\pi ft'}\mathrm{d}t' = \mathscr{F}\left\{\bar{R}_{XX}\right\} \tag{5.131}$$

Επομένως προκύπτει το συμπέρασμα ότι η PSD ενός τυχαίου σήματος $X_\tau(t)$ προκύπτει από τον μετασχηματισμό Fourier της μέσης τιμής $\bar{R}_{XX}$ της συνάρτησης αυτοσυσχέτισης R_{XX}:

$$\bar{R}_{XX}(t') = \lim_{\tau\to\infty}\frac{1}{\tau}\int_{-\tau/2}^{+\tau/2} R_{XX}(t'+t_2,t_2)\mathrm{d}t_2 \tag{5.132}$$

Για μία στατική τυχαία διαδικασία θα έχουμε $\bar{R}_{XX}(t') = R_{XX}(t')$ οπότε στην περίπτωση αυτή η PSD είναι απλώς ο μετασχηματισμός Fourier της συνάρτησης αυτοσυσχέτισης R_{XX}. Ας δούμε τώρα μερικά παραδείγματα υπολογισμού.

Παράδειγμα 5.4.4: **PSD του φέροντος σήματος**

Θα υπολογίσουμε την PSD $S_X(f)$, του φέροντος σήματος (5.73).

Λύση

Το τυχαίο σήμα που δίνεται από τον (5.73) είδαμε στο παράδειγμα 5.4.1 ότι έχει συνάρτηση αυτοσυσχέτισης που δίνεται από την σχέση

$$R_{XX}(t_1,t_2) = R_X X(t_1 - t_2) = \frac{A^2}{2}\cos(2\pi f_0(t_1 - t_2)) \tag{5.133}$$

Εφόσον η συνάρτηση αυτοσυσχέτισης εξαρτάται μόνο από την διαφορά $t' = t_1 - t_2$, πρόκειται για ένα στατικό τυχαίο σήμα, οπότε η PSD προκύπτει από τον μετασχηματισμό Fourier της $R_{XX}(t')$,

$$R_{XX}(t') = \frac{A^2}{2}\cos(2\pi f_0 t') \tag{5.134}$$

όπως έχουμε δει και στα προηγούμενα (παράδειγμα 4.6.2), ο μετασχηματισμός Fourier του συνημιτόνου δίνεται από την (4.48). Στην περίπτωση μας θα έχουμε

$$S_X(f) = \mathscr{F}\left\{R_{XX}(t')\right\} = \frac{A^2}{4}\left(\delta(f - f_0) + \delta(f + f_0)\right) \tag{5.135}$$

Η (5.135) μας δίδει την ζητούμενη PSD του σήματος.

Παράδειγμα 5.4.5: **PSD του ψηφιακού σήματος του παραδείγματος 5.4.2**

Θα θεωρήσουμε πάλι, το ψηφιακό σήμα της μορφής:

$$X(t) = \sum_{k=-\infty}^{+\infty} A_k p(t - kT_S) \tag{5.136}$$

όπου τα A_k θεωρούμε ότι είναι οι τιμές του ψηφιακού σήματος για $kT_S \leq t < (k+1)T_S$. Θα υπολογίσουμε την συνάρτηση αυτοσυσχέτισης $R_X X(t_1,t_2)$ θεωρώντας ότι τα A_k είναι ανεξάρτητες τυχαίες μεταβλητές με μέση τιμή $\mathbb{E}\{A_k\} = 0$ και έχουν την ίδια διακύμανση $\sigma_A^2 = \mathbb{E}\left\{A_k^2\right\}$. Θέλουμε να βρούμε την PSD του $X(t)$.

Λύση

Γνωρίζουμε από το παράδειγμα 5.4.2 ότι η συνάρτηση αυτοσυσχέτισης του σήματος

δίνεται από την σχέση:

$$R_{XX}(t_1,t_2) = \sigma_A^2 \sum_{k=-\infty}^{+\infty} p(t_1 - kT_S)p(t_2 - kT_S) \tag{5.137}$$

Για να βρούμε την PSD $S_X(f)$ πρέπει πρώτα να υπολογίσουμε την $\bar{R}_X X(t')$ που δίνεται από την σχέση (5.132)

$$\bar{R}_{XX}(t') = \lim_{\tau\to\infty} \frac{1}{\tau} \int_{-\tau/2}^{+\tau/2} R_{XX}(t'+t_2,t_2)\mathrm{d}t_2 = \lim_{\tau\to\infty} \frac{\sigma_A^2}{\tau} \int_{-\tau/2}^{+\tau/2} \sum_{k=-\infty}^{+\infty} p(t'+t_2-kT_S)p(t_2-kT_S)\mathrm{d}t_2 \tag{5.138}$$

Μπορούμε να δείξουμε ότι η συνάρτηση $R(t_2)$ όπου

$$R(t_2) = \sum_{k=-\infty}^{+\infty} p(t'+t_2-kT_S)p(t_2-kT_S) \tag{5.139}$$

είναι περιοδική με περίοδο T_S,

$$R(t_2+T_S) = \sum_{k=-\infty}^{+\infty} p(t'+t_2+T_S-kT_S)p(t_2+T_S-kT_S) = \sum_{m=-\infty}^{+\infty} p(t'+t_2-mT_S)p(t_2-mT_S) = R(t_2) \tag{5.140}$$

όπου έχουμε θέσει $m = k-1$. Εφόσον η $R(t_2)$ είναι περιοδική, η μέση τιμή της στο διάστημα $(-\infty,+\infty)$ θα είναι ίση με την μέση της τιμή στο $(0,T_S)$,

$$\bar{R}_{XX}(t') = \lim_{\tau\to\infty} \frac{1}{\tau} \int_{-\tau/2}^{+\tau/2} \sum_{k=-\infty}^{+\infty} p(t'+t_2-kT_S)p(t_2-kT_S)\mathrm{d}t_2 =$$
$$\lim_{\tau\to\infty} \frac{1}{\tau} \int_{-\tau/2}^{+\tau/2} R(t_2)\mathrm{d}t_2 = \frac{1}{T_S}\int_0^{T_S} R(t_2)\mathrm{d}t_2 = \frac{1}{T_S}\sum_{k=-\infty}^{+\infty}\int_0^{T_S} p(t'+t_2-kT_S)p(t_2-kT_S)\mathrm{d}t_2 \tag{5.141}$$

Θέτωντας $t = t_2 - kT_S$, μπορούμε να χρησιμοποιήσουμε τις ιδιότητες των ολοκληρωμάτων και να δείξουμε ότι:

$$\int_0^{T_S} p(t'+t_2-kT_S)p(t_2-kT_S)\mathrm{d}t_2 = \int_{kT_S}^{(k+1)T_S} p(t+t')p(t)\mathrm{d}t \tag{5.142}$$

οπότε:

$$\bar{R}_{XX}(t') = \frac{\sigma_A^2}{T_S}\sum_{k=-\infty}^{+\infty}\int_{kT_S}^{(k+1)T_S} p(t+t')p(t)\mathrm{d}t = \frac{\sigma_A^2}{T_S}\int_{-\infty}^{+\infty} p(t+t')p(t)\mathrm{d}t \tag{5.143}$$

Η (5.143) γράφεται:

$$\bar{R}_{XX}(t') = \frac{\sigma_A^2}{T_S}\int_{-\infty}^{+\infty} p(t+t')p(t)\mathrm{d}t = \frac{\sigma_A^2}{T_S}\int_{-\infty}^{+\infty} p(-t+t')p(t)\mathrm{d}t \tag{5.144}$$

Στην παραπάνω σχέση το ολοκλήρωμα είναι ο συγκερασμός του $p(-t')$ με το $p(t')$,

$$p(t') * p(-t') = \int_{-\infty}^{+\infty} p(t)p(-t+t')\mathrm{d}t \tag{5.145}$$

οπότε μπορούμε να γράψουμε:

$$\bar{R}_{XX}(t') = \frac{\sigma_A^2}{T_S} p(t') * p(-t') \tag{5.146}$$

Η PSD του σήματος είναι ο μετασχηματισμός Fourier του $R_{XX}(t')$,

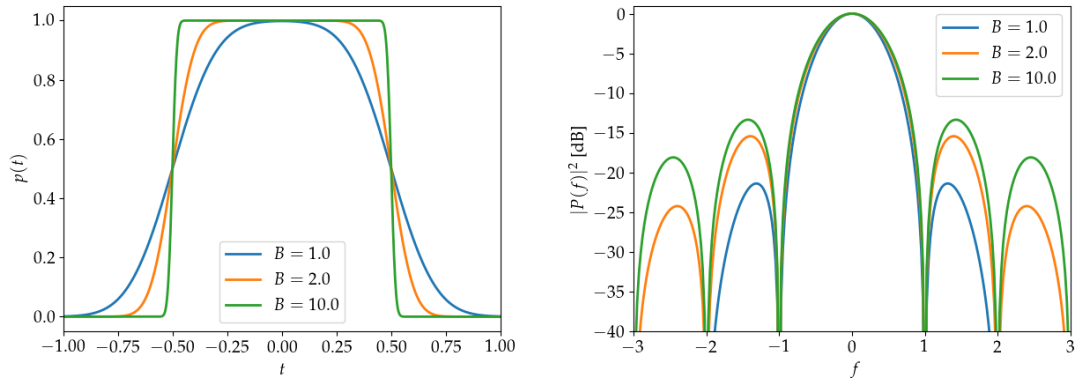
$$S_X(f) = \frac{\sigma_A^2}{T_S}\mathscr{F}\left\{p(t) * p(-t)\right\} = \frac{\sigma_A^2}{T_S}\mathscr{F}\left\{p(t')\right\}\mathscr{F}\left\{p(-t')\right\} = \frac{\sigma_A^2}{T_S}|P(f)|^2 \tag{5.147}$$

όπου $P(f) = \mathscr{F}\left\{p(t')\right\}$ είναι το φάσμα του $p(t')$ και έχουμε χρησιμοποιήσει την ιδιότητα του μετασχηματισμού Fourier, $\mathscr{F}\left\{p(-t')\right\} = P^*(f)$. Η (5.147) είναι η ζητούμενη PSD του σήματος $X(t)$.

Το παράδειγμα 5.4.5 και ειδικότερα η (5.147) είναι ιδιαίτερα χρήσιμα και στα παρακάτω. Δείχνουν ότι η κυματομορφή $p(t)$ καθορίζει την PSD και επομένως τα φασματικά χαρακτηριστικά του σήματος. Στο listing 5.11 υπολογίζουμε το $|P(f)|^2$, που είναι ανάλογο της PSD για διαφορετικές κυματομορφές $p(t)$ που παράγονται από

Στην εικόνα **??** δείχνουμε διαφορετικές μορφές του

```
import matplotlib.pyplot as plt
import numpy as np
from commlib import square_pulse, lti_system

plt.rcParams.update({
    "text.usetex": True,
    "font.family": "serif",
    "font.serif": ["Palatino"],
    "font.size" : 14,
    "lines.linewidth" : 2,
})

Tmax = 20
Np = 16384
T1 = 1
Bs = np.array([1, 2, 10]) / T1

t = np.linspace(-Tmax, Tmax, Np)
s = square_pulse(t = t, T1 = T1)

```



(a) (b)

Εικόνα 5.6: Η μορφή του παλμού $p(t)$ και το φάσμα $|P(f)|^2$.

```
plt.close('all')
plt.figure(1)

for B in Bs:
    H = lambda f : np.exp( -f ** 2.0 / 2 / B ** 2)
    S = lti_system( input_s = s, H = H )
    S.calc_output()
    p = S.output_s
    plt.figure(1)
    p.plot(ylabel = '$p(t)$', xlabel = '$t$', line_type = '-', label = '$B=' +
    str(B) + '$')
    plt.xlim([-1, 1])

    plt.figure(2)
    Pf_dB = 20 * np.log10(np.abs(p.spec))
    f = p.f

    plt.plot(f, Pf_dB,label = '$B=' + str(B) + '$')
    plt.xlabel('$f$')
    plt.ylabel('$|P(f)|^2$ [dB]', )
    plt.xlim([-3, 3])
    plt.ylim([-40, 1])
plt.figure(1)
plt.legend()

plt.figure(2)
plt.legend()
```

Listing 5.11: SXpulsetrain.py

5.5 Τυχαία σήματα και LTI συστήματα

Έχει ενδιαφέρον να δούμε τι συμβαίνει όταν ένα τυχαίο σήμα $X(t)$ περνάει από ένα γραμμικό και χρονικά αναλλοίωτο σύστημα (LTI). Η έξοδος του συστήματος θα δίνεται από την σχέση:

$$Y(t) = X(t) * h(t) = \int_{-\infty}^{+\infty} X(t)H(t-\tau)\mathrm{d}\tau \tag{5.148}$$

Για να υπολογίσουμε την συνάρτηση αυτοσυσχέτισης του $Y(t)$ θα έχουμε:

$$R_{YY}(t_1,t_2) = \mathbb{E}\{Y(t_1)Y(t_2)\} = \mathbb{E}\left\{\int_{-\infty}^{+\infty} X(\tau_1)h(t_1-\tau_1)\mathrm{d}\tau_1 \int_{-\infty}^{+\infty} X(\tau_2)h(t_2-\tau_2)\mathrm{d}\tau_2\right\} =$$
$$\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \mathbb{E}\{X(\tau_1)X(\tau_2)\}h(t_1-\tau_1)h(t_2-\tau_2)\mathrm{d}\tau_1\mathrm{d}\tau_2 =$$
$$\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} R_{XX}(\tau_1,\tau_2)h(t_1-\tau_1)h(t_2-\tau_2)\mathrm{d}\tau_1\mathrm{d}\tau_2 \tag{5.149}$$

Αν υποθέσουμε ότι το σήμα είναι στατικό, τότε ισχύει $R_{XX}(\tau_1,\tau_2) = R_{XX}(\tau_1-\tau_2)$. Επομένως η παραπάνω σχέση γράφεται:

$$R_{YY}(t_1,t_2) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} R_{XX}(\tau_1-\tau_2)h(t_1-\tau_1)h(t_2-\tau_2)\mathrm{d}\tau_1\mathrm{d}\tau_2 \tag{5.150}$$

Στο εσωτερικό ολοκλήρωμα θέτουμε $\tau = \tau_1 - \tau_2$, οπότε:

$$R_{YY}(t_1,t_2) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} R_{XX}(\tau)h(t_1-\tau+\tau_2)h(t_2-\tau_2)\mathrm{d}\tau\mathrm{d}\tau_2 =$$
$$\int_{-\infty}^{+\infty} h(t_2-\tau_2)\left[\int_{-\infty}^{+\infty} R_{XX}(\tau)h(t_1-\tau+\tau_2)\mathrm{d}\tau\right]\mathrm{d}\tau_2 \tag{5.151}$$

Αν θυμηθούμε τον ορισμό του συγκερασμού στην (**??**), τότε το εσωτερικό ολοκλήρωμα γράφεται και ως εξής:

$$z(t_1+\tau_2) = \int_{-\infty}^{+\infty} R_{XX}(\tau)h(t_1-\tau+\tau_2)\mathrm{d}\tau = R_{XX}(t_1+\tau_2) * h(t_1+\tau_2) \tag{5.152}$$

και έχουμε:

$$R_{YY}(t_1,t_2) = \int_{-\infty}^{+\infty} h(t_2-\tau_2)z(t_1+\tau_2)\mathrm{d}\tau_2 \tag{5.153}$$

Αν κάνουμε μία αλλαγή μεταβλητής $\tau' = t_2 - \tau_2$ τότε το παραπάνω ολοκλήρωμα πάλι θα γραφεί με την μορφή συγκερασμού:

$$R_{YY}(t_1,t_2) = \int_{-\infty}^{+\infty} h(-\tau')z(t_1-t_2-\tau')\mathrm{d}\tau' = z(t_1-t_2) * h(-(t_1-t_2)) =$$
$$R_{XX}(t_1-t_2) * h(t_1-t_2) * h(-(t_1-t_2)) \tag{5.154}$$

Από την παραπάνω σχέση επομένως προκύπτει ότι το $R_{YY}(t_1,t_2)$ εξαρτάται μόνο από το t_1-t_2 οπότε το Y είναι ένα στατικό σήμα και έχουμε:

$$R_{YY}(\tau) = R_{XX}(\tau) * h(\tau) * h(-\tau) \tag{5.155}$$

Αν θέλουμε να υπολογίσουμε την PSD της εξόδου $S_Y(f)$ τότε θα πρέπει να υπολογίσουμε τον μετασχηματισμό Fourier της παραπάνω σχέσης,

$$S_Y(f) = \mathscr{F}\{R_{XX}(\tau) * h(\tau) * h(-\tau)\} = \mathscr{F}\{R_{XX}(\tau)\}\mathscr{F}\{h(\tau)\}\mathscr{F}\{h(-\tau)\} \tag{5.156}$$

Αν $H(f) = \mathscr{F}\{h(\tau)\}$ είναι η συνάρτηση μεταφοράς του συστήματος και $S_X(f) = \mathscr{F}\{R_{XX}(\tau)\}$ είναι η PSD του σήματος εισόδου, θα έχουμε:

$$S_Y(f) = S_X(f)H(f)H^*(f) = S_X(f)|H(f)|^2 \tag{5.157}$$

όπου χρησιμοποιήσαμε το γεγονός ότι ο μετασχηματισμός Fourier του $h(-\tau)$ είναι το συζυγές $H^*(f)$ του $H(f)$.

Τυχαία σήματα μέσα από ένα LTI σύστημα

Όταν ένα στατικό τυχαίο σήμα $X(t)$ τοποθετείται στην είσοδο ενός LTI συστήματος με κρουστική απόκριση $h(t)$, τότε η έξοδος του $Y(t)$ είναι επίσης στατική και η συνάρτηση αυτοσυσχέτισης δίνεται από τον διπλό συγκερασμό:

$$R_{YY}(\tau) = R_{XX}(\tau) * h(\tau) * h(-\tau) \tag{5.158}$$

ενώ η PSD εξόδου $S_Y(f)$ συνδέεται με την PSD εισόδου $S(f)$ βάσει της σχέσης:

$$S_Y(f) = S_X(f)H(f)H^*(f) = S_X(f)|H(f)|^2 \tag{5.159}$$

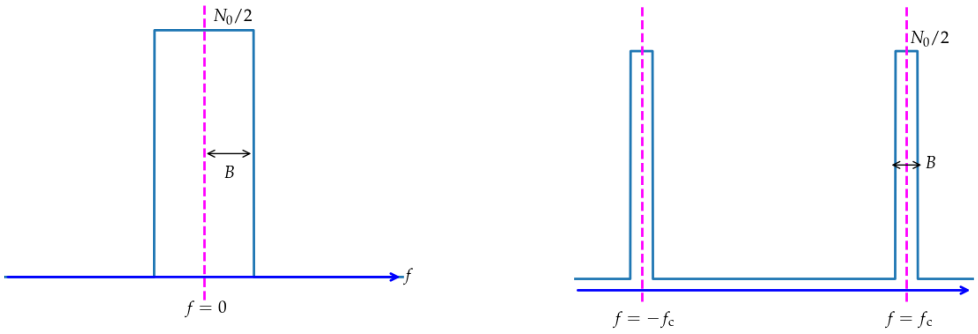
5.6 Θόρυβος

Στις επικοινωνίες θα δούμε ότι στον δέκτη πάντα προστίθεται ένα τυχαίο σήμα $n(t)$ που ονομάζεται θόρυβος. Η φασματική πυκνότητα ισχύος του $n(t)$, $S_n(f)$ παίζει πολύ σημαντικό ρόλο στις επιδόσεις του συστήματος. Συχνά θεωρούμε ότι η φασματική πυκνότητα ισχύος είναι σταθερή στο διάστημα των συχνοτήτων που μας ενδιαφέρει και γράφουμε

$$S_n(f) = \frac{N_0}{2} \tag{5.160}$$

Στην παραπάνω εξίσωση το N_0 ονομάζεται συχνά *φασματική πυκνότητα ισχύος θορύβου*. Θεωρούμε ότι ο θόρυβος $n(t)$ είναι ένα πραγματικό σήμα, τότε όπως είδαμε και στην ενότητα 4.7, το φάσμα του θα εκτείνεται και στις θετικές και στις αρνητικές συχνότητες. Στην εικόνα 5.7, δείχνουμε δύο παραδείγματα φάσματος λευκού θορύβου ο οποίος όμως έχει περιοριστεί σε ένα μικρό εύρος ζώνης B στις χαμηλές συχνότητες (εικόνα 5.7a) ή γύρω από ένα μικρό εύρος ζώνης στις υψηλές συχνότητες (εικόνα 5.7b). Και στις δύο περιπτώσεις είναι εύκολο να δούμε ότι η αναμενόμενη ισχύς του θορύβου $\mathbb{E}\{n^2(t)\}$ θα είναι το εμβαδό $\int S_X(f)\mathrm{d}f$ της $S_X(f) = N_0/2$ μέσα στο εύρος των συχνοτήτων που μας ενδιαφέρουν, δηλαδή η ισχύς του $P(t)$ δίνεται από την:

$$P(t) = \mathbb{E}\{n^2(t)\} = N_0 B \tag{5.161}$$



(a) (b)

Εικόνα 5.7: Λευκός θόρυβος στις χαμηλές και στις υψηλές συχνότητες.

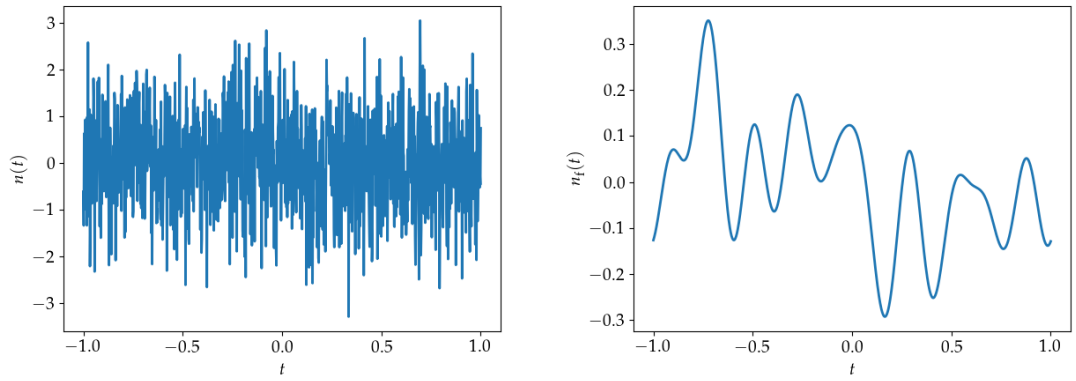
Αν θεωρήσουμε ότι ο θόρυβος είναι στατικός, τότε μπορούμε να υπολογίσουμε την συνάρτηση αυτοσυσχέτισης του σήματος:

$$R_{nn}(t_1,t_2) = R_{nn}(t_1-t_2) = \mathbb{E}\{n(t_1)n(t_2)\} = \frac{N_0}{2}\delta(t_1-t_2) \tag{5.162}$$

όπου χρησιμοποιήσαμε ότι ο μετασχηματισμός Fourier μίας σταθερά είναι μία συνάρτηση $\delta(t)$.

Εξετάζοντας την (5.162), προκύπτουν δύο ενδιαφέροντα συμπεράσματα. Το πρώτο είναι ότι εάν $t_1 \neq t_2$ τότε θα έχουμε $R_{nn}(t_1,t_2) = 0$ και επομένως η συνάρτηση αυτοσυσχέτισης είναι μηδέν. Στην περίπτωση όπου τα $n(t)$ έχουν κανονική κατανομή (κάτι που θα υποθέσουμε και παρακάτω στις επόμενες ενότητες), τότε έπεται ότι το $n(t_1)$ είναι ανεξάρτητο του $n(t_2)$ όπως είδαμε και στην ενότητα 5.3.1. Επομένως αν παρατηρούμε το θόρυβο την στιγμή $t = t_1$ και $t_2 > t_1$ τότε η μελλοντική τιμή του θορύβου $n(t_2)$ δεν εξαρτάται καθόλου από την τιμή του σε μία προηγούμενη χρονική στιγμή όπως η t_1.

```
from commlib import signal, lti_system
import matplotlib.pyplot as plt
import numpy as np

B = 10
H = lambda f: (np.abs(f)<=B/2).astype(float)

plt.rcParams.update({
    "text.usetex": True,
    "font.family": "serif",
    "font.serif": ["Palatino"],
    "font.size" : 14,
    "lines.linewidth" : 2,
})
Tmax = 1
Nsamples = 1024
t = np.linspace(-Tmax, Tmax, Nsamples)

# white noise samples
```



(a) (b)

Εικόνα 5.8: (a) Λευκός και (b) χρωματισμένος θόρυβος.

```
nsamples = np.random.randn( Nsamples )
n = signal(t = t, samples = nsamples)

plt.close('all')
plt.figure(1)
n.plot(xlabel = '$t$', ylabel = '$n(t)$', line_type = '-')

# filtered noise samples
S = lti_system(input_s = n, H = H)
S.calc_output()
n_filtered = S.output_s
plt.figure(2)
n_filtered.plot(xlabel = '$t$', ylabel = '$n_\mathrm{f}(t)$', line_type = '-')
```

Listing 5.12: noises.py

Στην εικόνα 5.8 δείχνουμε παραδείγματα σημάτων θορύβου που παράγονται χρησιμοποιώντας το listing 5.12. Στην περίπτωση της εικόνας 5.8a, τα δείγματα του θορύβου έχουν παραχθεί απευθείας από την `numpy.random.randn` η οποία παράγει ανεξάρτητα μεταξύ τους δείγματα με μέση τιμή μηδέν και επομένως η συνάρτηση αυτοσυσχέτισης τους θα είναι μηδέν, όπως δηλαδή και στην περίπτωση του λευκού θορύβου. Στην περίπτωση της εικόνας 5.8b, περνάμε τα δείγματα του θορύβου από ένα σύστημα το οποίο έχει συνάρτηση μεταφοράς

$$H(f) = \begin{cases} 1, & |f| < B/2 \\ 0, & \text{διαφορετικά} \end{cases} \tag{5.163}$$

Με τον τρόπο αυτό, η PSD θα είναι κόβουμε τις υψηλές συχνότητες και φιλτράρονται οι πολύ γρήγορες μεταβάσεις των δειγμάτων της εικόνας 5.8a και δημιουργείται μία συσχέτιση μεταξύ τους όπως φαίνεται στην εικόνα 5.8b. Θυμηθείτε η PSD δίνεται από την σχέση (5.159) και επομένως θα έχουμε:

$$S_Y(f) = \begin{cases} S_X(f), & |f| < B/2 \\ 0, & \text{διαφορετικά} \end{cases} \tag{5.164}$$

Σύμφωνα με την παραπάνω σχέση, η PSD του σήματος $Y(t)$ στην έξοδο του συστήματος ταυτίζεται με την PSD στην είσοδο όταν η συχνότητα είναι μικρότερη του B. Για συχνότητες μεγαλύτερες του B, το $S_Y(f)$ είναι μηδέν. Για τον λόγο αυτό παρατηρείται η συμπεριφορά στην εικόνα 5.8b.

Ένα δεύτερο συμπέρασμα το οποίο αρχικά μπορεί να μας ξενίσει είναι ότι όταν $t_1 = t_2$ τότε θα έχουμε $R_{nn}(t_1,t_1) = \mathbb{E}\left\{n^2(t_1)\right\} = \frac{N_0}{2}\delta(0)$ και αν θυμηθούμε ότι το $\delta(t)$ τείνει στο άπειρο (∞) όταν το t τείνει στο μηδέν οπότε και το $\mathbb{E}\left\{n^2(t_1)\right\}$ θα είναι άπειρο, $\mathbb{E}\left\{n^2(t_1)\right\} \to \infty$. Επομένως η στιγμιαία ισχύς του $n(t)$, $P(t) = \mathbb{E}\left\{n^2(t)\right\}$ είναι και αυτή άπειρη. Πως γίνεται αυτό; Ας ξαναδούμε την (5.161). Όταν το εύρος ζώνης που κοιτάμε τον θόρυβο είναι B τότε η ισχύς του είναι ανάλογη του B. Αν κοιτάζαμε τον θόρυβο σε όλη τον άξονα των συχνοτήτων ($B \to \infty$) επομένως θα έχουμε και $P(t) \to \infty$. Στην πράξη βέβαια δεν μπορούμε να μετρήσουμε το θόρυβο σε ολόκληρο τον άξονα των συχνοτήτων. Για παράδειγμα αν ο θόρυβος είναι ένα σήμα τάσης, τότε ο παλμογράφος που θα χρησιμοποιήσουμε θα έχει μία απόκριση σαν αυτήν της σχέσης (5.163), οπότε θα μπορεί να "ακούσει" ένα εύρος συχνοτήτων $f \in [-B,B]$ όπου για κοινούς παλμογράφους συνήθως το B είναι της τάξης των μερικών MHz ενώ για πιο εξελιγμένους παλμογράφους (και πιο ακριβούς) το B μπορεί να φτάσει τα μερικές δεκάδες GHz. Επομένως είμαστε περιορισμένοι σε ένα φασματικό εύρος όπως δείχνει η εικόνα 5.7a ή κάποιες άλλες φορές η εικόνα 5.7b και τελικά η ισχύς του θορύβου είναι πεπερασμένη και όχι άπειρη.

5.6.1 Τι μάθαμε

Στο κεφάλαιο αυτό ασχοληθήκαμε καταρχήν με την έννοια της τυχαίας μεταβλητής της οποίας την τιμή δεν γνωρίζουμε αλλά μπορούμε να υπολογίσουμε την πιθανότητα η μεταβλητή να μία τιμή, στην περίπτωση των διακριτών τυχαίων μεταβλητών ή να λάβει τιμή εντός ενός διαστήματος τιμών στην περίπτωση των συνεχών τυχαίων μεταβλητών. Για τις συνεχείς μεταβλητές είδαμε τον βασικό ρόλο που παίζει η PDF και πως υπολογίζονται οι αναμενόμενες τιμές συναρτήσεων τυχαίων μεταβλητών. Επίσης συζητήσαμε την συσχέτιση μεταξύ δύο μεταβλητών και πως αυτή μπορεί να αποτιμηθεί. Είδαμε παραδείγματα χρήσιμων τυχαίων μεταβλητών όπως η κανονική και η ομοιομόρφη τυχαία μεταβλητή. Στη συνέχεια είδαμε πως περιγράφουμε τα τυχαία σήματα, τα οποία τα συναντάμε πολύ συχνά στα συστήματα επικοινωνιών. Ορίσαμε την συνάρτηση αυτοσυσχέτισης και είδαμε πως αυτή καθορίζει τις ιδιότητες του τυχαίου σήματος στο πεδίο των συχνοτήτων. Τα θεωρητικά και προγραμματιστικά εργαλεία που αναπτύξαμε σε αυτό το κεφάλαιο θα χρησιμοποιηθούν στα επόμενα για την ανάλυση των επιδόσεων των συστημάτων επικοινωνιών στα επόμενα κεφάλαια.

6. Διαμόρφωση Κατά Πλάτος

6.1 Εισαγωγή

Στο κεφάλαιο αυτό, θα εξετάσουμε πως γίνεται η μετάδοση του σήματος χρησιμοποιώντας *διαμόρφωση πλάτους*. Η τεχνική αυτή χρησιμοποιείται αρκετά συχνά στην πράξη και συνίσταται στην αποτύπωση του ψηφιακού σήματος στο πλάτος ενός φυσικού μεγέθους, όπως η ηλεκτρική τάση ή το ηλεκτρικό και μαγνητικό πεδίο ενός ηλεκτρομαγνητικού πεδίου. Ανάλογα με την περιοχή του φάσματος που χρησιμοποιείται μπορούμε να διαχωρίσουμε τα συστήματα σε *βασικής ζώνης* (baseband), όταν το φάσμα του σήματος μας είναι συγκεντρωμένο κοντά στη συχνότητα $f = 0$ ή σε συστήματα *ζωνοπεράτα* (passband) όπου το φάσμα είναι "μαζεμένο" γύρω από μία κεντρική συχνότητα $f = f_c$ και το εύρος ζώνης του σήματος B είναι πολύ μικρότερο από το f_c, $B << f_c$. Στην εικόνα 6.1, δείχνουμε τυπικά φάσματα για ένα σήμα βασικής ζώνης και ένα ζωνοπερατό σήμα.

Έχει μία ιδιαίτερη σημασία να σχολιάσουμε την συμμετρία που παρατηρούμε στα φάσματα αυτά. Αν θυμηθούμε τον ορισμό του μετασχηματισμού Fourier από την (4.24)

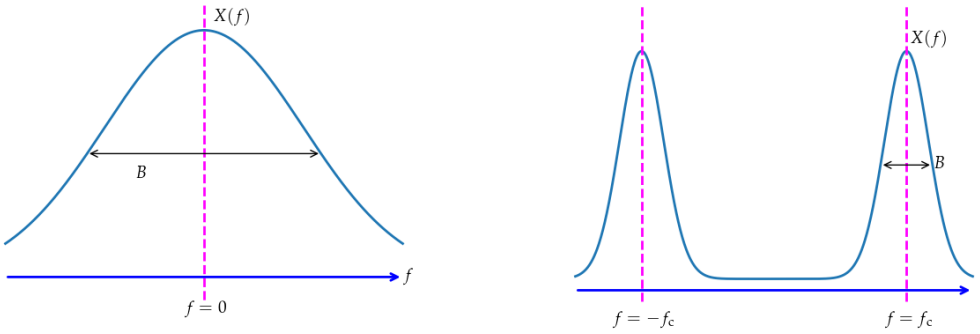
$$X(f) = \mathscr{F}\{x(t)\} = \int_{-\infty}^{+\infty} x(t) \mathrm{e}^{-j2\pi ft} \mathrm{d}t \tag{6.1}$$

Αν θεωρήσουμε ένα πραγματικό σήμα $x(t)$ και προσπαθήσουμε να υπολογίσουμε το μιγαδικό συζυγές $X^*(-f)$ του $X(-f)$ θα έχουμε:

$$X^*(-f) = \left(\int_{-\infty}^{+\infty} x(t) \mathrm{e}^{j2\pi ft} \mathrm{d}t \right)^* = \int_{-\infty}^{+\infty} \left[x(t) \mathrm{e}^{j2\pi ft} \mathrm{d}t \right]^* \mathrm{d}t = \int_{-\infty}^{+\infty} x^*(t) \mathrm{e}^{-j2\pi ft} \mathrm{d}t = \int_{-\infty}^{+\infty} x(t) \mathrm{e}^{-j2\pi ft} \mathrm{d}t = X(f) \tag{6.2}$$

Επομένως προκύπτει ότι το $X^*(-f) = X(f)$ και εφόσον δύο μιγαδικοί αριθμοί που είναι συζηγείς έχουν το ίδιο μέτρο θα ισχύει

$$|X(f)| = |X(-f)| \tag{6.3}$$



(a) (b)

Εικόνα 6.1: Παραδείγματα φασμάτων σημάτων: (a) βασικής ζώνης και (b) ζωνοπερατό.

Δηλαδή το μέτρο του φάσματος $|X(f)|$ θα είναι συμμετρικό γύρω από το $f = 0$. Για το λόγω επομένως αυτόν, έχουμε ζωγραφίσει τα φάσματα να είναι συμμετρικά γύρω από το $f = 0$. Παρατηρούμε στην εικόνα 6.1 την διαφορετική μορφή των φασμάτων. Τα σήματα βασικής ζώνης (εικόνα 6.1a) χρησιμοποιούνται συνήθως στην ενσύρματη μετάδοση. Για παράδειγμα, όταν μεταδίδουμε ένα σήμα πάνω από μία δισύρματη γραμμή μεταφοράς (όπως π.χ. το παλιό τηλεφωνικό καλώδιο) τότε συνήθως τα δεδομένα αποτυπώνονται σε ένα σήμα βασικής ζώνης. Όταν μεταδίδουμε το σήμα ασύρματα, θα πρέπει να χρησιμοποιήσουμε ζωνοπερατά σήματα. Π.χ. στο WiFi χρησιμοποιούμε συχνότητες κοντά στο $f_c = 2.4$ GHz και στο $f_c = 5$ GHz με διαθέσιμο εύρος ζώνης $B \approx 100$ MHz και $B \approx 50$ MHz αντίστοιχα.

Στην περίπτωση της διαμόρφωσης πλάτους βασικής ζώνης, εφαρμόζουμε το ψηφιακό σήμα $x(t)$ που είδαμε στην παράγραφο 3.6 και συγκεκριμένα στην σχέση (3.15) απευθείας στο κανάλι, π.χ. ως τάση στην δισύρματη γραμμή μεταφοράς,

$$x(t) = \sum_{k=0}^{N} x_k p(t - kT_S) \tag{6.4}$$

Στην (6.4), το x_k είναι το πλάτος του σήματος που αντιστοιχεί στο $k^{\text{οστό}}$ σύμβολο προς μετάδοση, T_S είναι η διάρκεια του κάθε συμβόλου. Στις επόμενες παραγράφους θα εξετάσουμε λίγο πιο συστηματικά την μετάδοση της πληροφορίας με αυτό τον τρόπο που ονομάζεται *Διαμόρφωση Παλμού κατά Πλάτος* (pulse amplitude modulation - PAM) στη βασική ζώνη αλλά και με την μετάδοση σε υψηλότερες συχνότητες (ζωνοπεράτά σήματα).

6.2 Κωδικοποίηση

Η πληροφορία που θέλουμε να μεταδώσουμε συνήθως έχει την μορφή δυαδικών bit b_m τα οποία μπορούν να πάρουν τιμές μηδέν ή ένα. Οπότε τα συστήματα μας θα πρέπει να μεταδώσουν μηνύματα της μορφής:

$$\mathbf{b} = \{b_0, b_1, \cdots, b_p, \cdots\} \tag{6.5}$$

Θα πρέπει με κάποιον τρόπο να αντιστοιχήσουμε στα $\mathbf{b}$ ένα σύνολο από σύμβολα

$$\mathbf{x} = \{x_0, x_1, \cdots, x_k, \cdots\} \tag{6.6}$$

Ένας προφανής τρόπος να γίνει αυτή η αντιστοίχιση είναι κάθε bit b_p να αντιστοιχηθεί σε ένα ακριβώς σύμβολο x_p, για παράδειγμα εάν το $b_p = 1$ να θέσουμε $x_p = 1$ και εάν $b_p = 0$ να θέσουμε $x_p = -1$,

$$\begin{aligned} \{0\} &\rightarrow -1 \\ \{1\} &\rightarrow +1 \end{aligned} \tag{6.7}$$

Έτσι αντιστοιχούμε *ένα bit ανά σύμβολο*. Ωστόσο υπάρχουν και άλλες επιλογές που συνήθως οδηγούν σε καλύτερα αποτελέσματα. Θα μπορούσαμε να ομαδοποιήσουμε τα bit σε ομάδες των m διαδοχικών bit. Έτσι π.χ. αν $m = 3$, η πρώτη ομάδα είναι η $\{b_0, b_1, b_2\}$, η δεύτερη ομάδα είναι η $\{b_3, b_4, b_5\}$ κ.ο.κ. Αν θεωρήσουμε m bits υπάρχουν

$$= 2^m \tag{6.8}$$

διαφορετικοί συνδυασμοί που μπορούν να προκύψουν. Για παράδειγμα αν $m = 2$ τότε οι πιθανοί συνδυασμοί μίας ομάδας δύο διαδοχικών bit είναι $\{00, 01, 10, 11\}$ οπότε έχουμε $M = 4 = 2^2$ διαφορετικούς συνδυασμούς. Θα μπορούσαμε να αναθέσουμε σε κάθε συνδυασμό ένα διαφορετικό σύμβολο. Για παράδειγμα,

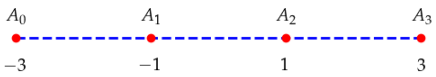
$$\begin{aligned} \{00\} &\rightarrow -3 \\ \{01\} &\rightarrow -1 \\ \{10\} &\rightarrow +1 \\ \{11\} &\rightarrow +3 \end{aligned} \tag{6.9}$$

Η αντιστοίχιση συνδυασμών bit σε πλάτη συμβόλων όπως στην (6.7) και στην (6.9), ονομάζεται *κωδικοποίηση*. Η κωδικοποιήσεις αυτές έχουν το χαρακτηριστικό ότι το σύνολο των πιθανών τιμών A_n που μπορεί να λάβει ένα σύμβολο είναι ισαπέχουσες. Έτσι για παράδειγμα στην (6.9) έχουμε $M = 4$ πιθανές τιμές $A_0 = -3, A_1 = -1, A_2 = +1$ και $A_3 = +3$ για όλα τα x_k. Στην εικόνα (6.2) δείχνουμε τα 4 πιθανά σύμβολα από όπου εύκολα προκύπτει ότι τα διαδοχικά σύμβολα είναι ισαπέχοντα. Επίσης τα πλάτη A_k είναι συμμετρικά γύρω από το μηδέν.

Υπάρχει ωστόσο ένα πρόβλημα με την κωδικοποίηση στην (6.9): τα σφάλματα στις επικοινωνίες που μας οδηγήσουν σε εσφαλμένη αποκωδικοποίηση εξαιτίας του θορύβου, όπως θα δούμε και παρακάτω. Έτσι εμείς μπορεί να θέλουμε να μεταδώσουμε το $\{00\}$ οπότε στέλνουμε το πλάτος -3 αλλά ο θόρυβος που προστίθεται μπορεί να μας φέρει πιο κοντά στο -1 και ο δέκτης να θεωρήσει ότι έλαβε αυτό το σύμβολο και όχι το -3. Οπότε θα αποκωδικοποιήσει τα bit $\{01\}$ και όχι το $\{00\}$ και θα λάβει χώρα ένα σφάλμα καθώς το δεύτερο bit θα αποκωδικοποιηθεί ως 1 και όχι ως 0. Τα πράγματα είναι χειρότερα όταν θέλουμε να εκπέμψουμε το $\{01\}$ οπότε στέλνουμε το -1, ο θόρυβος μας στέλνει πιο κοντά στο $+1$ και ο δέκτης εσφαλμένα καταλαβαίνει ότι στείλαμε το $\{10\}$. Σε αυτή την περίπτωση έχουμε δύο σφάλματα στα bit του δέκτη: το πρώτο bit αντί για 0 αποκωδικοποιείται ως 1 ενώ για το δεύτερο bit αποκωδικοποιείται ως 0 ενώ στέλνουμε 1.

Για να έχουμε μικρό αριθμό σφαλμάτων, θα πρέπει να ελαττώσουμε τον αριθμό των εσφαλμένων bit όταν ο θόρυβος μας "πετάει" σε γειτονικά σύμβολα. Για να το κάνουμε αυτό θα πρέπει να επιλέξουμε μία εναλλακτική κωδικοποίηση όπως φαίνεται παρακάτω

$$\begin{aligned} \{00\} &\rightarrow -3 \\ \{01\} &\rightarrow -1 \\ \{11\} &\rightarrow +1 \\ \{10\} &\rightarrow +3 \end{aligned} \tag{6.10}$$



Εικόνα 6.2: Τα 4 πιθανά πλάτη των συμβόλων του PAM οταν $M = 4$.

Στην κωδικοποίηση (6.10) έχουμε αλλάξει την σειρά των δύο τελευταίων συνδυασμών bit στην (6.9) από $\{10\},\{11\}$ σε $\{11\},\{10\}$. Με τον τρόπο αυτό είναι φανερό ότι πλέον δεν θα έχουμε δύο σφάλματα bit όταν πηγαίνουμε από το -1 στο $+1$ αλλά ένα. Επίσης αν κοιτάξουμε προσεκτικά, σε καμία περίπτωση δεν έχουμε σφάλματα όταν πηγαίνουμε σε γειτονικό σύμβολο. Π.χ. αν πάμε από το -3 στο -1 η ακολουθία bit που αποκωδικοποιούμε θα είναι $\{01\}$ ενώ θα έπρεπε να είναι $\{00\}$, οπότε πάλι κάνουμε σφάλμα σε ένα μόνο bit.

Υπάρχει ένας συστηματικός τρόπος να επιλέγουμε την κατάλληλη κωδικοποίηση που οδηγεί σε ένα μόνο σφάλμα bit όταν πηγαίνουμε σε γειτονικό σύμβολο, ο κώδικας *Gray*. Για να φτιάξουμε τον κώδικα Gray ακολουθούμε έναν αναδρομικό τρόπο κατασκευής σύμφωνα με τον οποίο ο κώδικας Gray $\mathbf{G}_m$ με m bits προκύπτει από τον κώδικα Gray $\mathbf{G}_{m-1}$ με $m-1$ bits. Ας θεωρήσουμε για παράδειγμα τον κώδικα Gray $\mathbf{G}_2$ στην (6.10)

$$\mathbf{G}_2 = [00, 01, 11, 10] \tag{6.11}$$

Στην ουσία το $\mathbf{G}_2$ περιέχει τους διαδοχικούς συνδυασμούς των συνδυασμών της κωδικοποίησης στην (6.11). Στην γενική περίπτωση το $\mathbf{G}_m$ είναι ένα διάνυσμα που περιέχει όλους τους συνδυασμούς με την σειρά που ανατίθενται σε διαδοχικά σύμβολα. Αν ξέρουμε το $\mathbf{G}_m$ επομένως εύκολα αναθέτουμε συνδυασμούς σε διαδοχικά σύμβολα όπως στην (6.11).

Για να φτιάξουμε το $\mathbf{G}_3$ από το $\mathbf{G}_2$ προσθέτουμε καταρχήν ένα ο μπροστά από τους συνδυασμούς του $\mathbf{G}_2$. Έχουμε επομένως τους συνδυασμούς:

$$[000, 001, 011, 010] \tag{6.12}$$

Προς το παρόν έχουμε 4 συνδυασμούς ενώ χρειαζόμαστε $2^3 = 8$ συνδυασμούς. Για να τους φτιάξουμε παίρνουμε τους συνδυασμούς του $\mathbf{G}_2$, τους αντιστρέφουμε, δηλαδή θεωρούμε με την σειρά $[10, 11, 01, 00]$ και προσθέτουμε το 1 μπροστά οπότε φτιάχνουμε την ακολουθία:

$$[110, 111, 101, 100] \tag{6.13}$$

Ενώνοντας τις δύο ακολουθίες έχουμε:

```
In [3]: g = gray_code(4)

In [4]: g
Out[4]:
['0000',
 '0001',
 '0011',
 '0010',
 '0110',
 '0111',
 '0101',
 '0100',
 '1100',
 '1101',
 '1111',
 '1110',
 '1010',
 '1011',
 '1001',
 '1000']
```

Εικόνα 6.3: Οι συνδυασμοί του κώδικα Gray για $m = 4$.

$$[000, 001, 011, 010, 110, 111, 101, 100] \tag{6.14}$$

που αν προσέξουμε έχει την επιθυμητή ιδιότητα, δηλαδή οι διαδοχικοί συνδυασμοί απέχουν κατά ένα ακριβώς bit. Θέτουμε επομένως

$$\mathbf{G}_3 = [000, 001, 011, 010, 110, 111, 101, 100] \tag{6.15}$$

```
54 def gray_code(m):
55
56     if m==1:
57         g = ['0','1']
58
59     elif m>1:
60         gs = gray_code(m-1)
61         gsr = gs[::-1]
62         gs0 = ['0' + x for x in gs]
63         gs1 = ['1' + x for x in gsr]
64         g= gs0 + gs1
65     return g
```

Listing 6.1: Υπολογισμός κώδικα Gray

Στο listing 6.1 δείχνουμε τον τρόπο υπολογισμού του κώδικα Gray χρησιμοποιώντας την αναδρομική σχέση που συζητήσαμε. Στην τετριμμένη περίπτωση όπου $m = 1$, ο κώδικας Gray είναι απλά το $\mathbf{G}_1 = [0, 1]$. Διαφορετικά εάν $m > 1$ τότε υπολογίζουμε τους συνδυασμούς του $\mathbf{G}_m$ από τους συνδυασμούς του $\mathbf{G}_{m-1}$ στους οποίους προσθέτουμε μπροστά το 0 ενώ προσθέτουμε το 1 στους συνδυασμούς του $\mathbf{G}_{m-1}$ σε ανεστραμμένη όμως σειρά.

Στην εικόνα 6.3 δείχνουμε το αποτέλεσμα της `graycode` για $m = 4$. Οι διαδοχικοί συνδυασμοί που παράγονται είναι φανερό ότι διαφέρουν κατά ένα ακριβώς bit.

6.3 Αστερισμοί Συμβόλων

Στην ενότητα 6.2, ασχοληθήκαμε με την κωδικοποίηση, δηλαδή την αντιστοίχιση των συνδυασμών bit σε πλάτη συμβόλων A_k. Στην ενότητα αυτή θα ασχοληθούμε λίγο περισσότερο

με τα A_k. Το σύνολο των δυνατών A_k συχνά ονομάζεται *αστερισμός συμβόλων*. Στην περίπτωση του PAM με $m = 2$ bit ανά σύμβολο έχουμε όπως είδαμε στην ενότητα 6.2, $M = 2^m = 4$ δυνατές τιμές των A_k και η εικόνα 6.2 δείχνει μία επιλογή για τις τιμές των A_k η οποία είναι συμμετρική γύρω από το 0 και τα διαδοχικά πλάτη ισαπέχουν, δηλαδή θα έχουμε πάντα:

$$A_{k+1} - A_k = 2\beta \tag{6.16}$$

Το 2β είναι η απόσταση μεταξύ δύο διαδοχικών συμβόλων Ένα σύστημα PAM με πλήθος συμβόλων M ονομάζεται *M-PAM* και για το 4-PAM στην εικόνα 6.2 έχουμε $\beta = 1$. Στην γενικότερη περίπτωση όπου το πλήθος των συμβόλων είναι $M = 2^m$ μπορούμε να καταλήξουμε σε μία σχέση για τα A_k στην περίπτωση του M-PAM. Από την (6.16) προκύπτει ότι:

$$A_k = A_0 + 2k\beta \tag{6.17}$$

Από την παραπάνω σχέση και αν υποθέσουμε ότι μετρούμε τα A_k από $k = 0$ μέχρι το $k = M - 1$ προκύπτει ότι $A_{M-1} = A_0 + 2\beta(M-1)$. Εφόσον τα A_k είναι συμμετρικά ως προς το 0 θα πρέπει $A_{M-1} = -A_0$ και επομένως θα έχουμε

$$-A_0 = A_0 + 2\beta(M-1) \Rightarrow A_0 = -\beta(M-1) \tag{6.18}$$

και χρησιμοποιώντας την (6.17) προκύπτει ότι

$$A_k = -\beta(M-1) + 2k\beta = (2k - M + 1)\beta \tag{6.19}$$

Η (6.19) μας παρέχει μία σχέση για να υπολογίζουμε τα A_k για το M-PAM στην περίπτωση όπου τα πλάτη είναι συμμετρικά και διαδοχικά ισαπέχοντα.

6.4 Η κλάση constellation

Θα χρησιμοποιήσουμε μία γενική κλάση, την `constellation` για να περιγράψουμε τους αστερισμούς των συστημάτων και στη συνέχεια θα υλοποιήσουμε την `pam_constellation` η οποία κληρονομεί την `constellation` και περιγράφει τον αστερισμό του M-PAM. Στο listing 6.2 δείχνουμε την κλάση `constellation`.

```
352 class constellation:
353
354     def __init__(self, title = None):
355         self.map = {}
356         self.bits = []
357         self.bits_str = []
358         self.symbols = []
359         self.title = title
360         self.m = None
361
362     def avg_power(self):
363         return np.mean( np.abs(self.symbols) ** 2.0)
364
365     def plot(self, figure_no = None, plot_type = 'o'):
366
367         if figure_no is None:
368             plt.figure()
369         else:
```

```
370             plt.figure(figure_no)
371
372         cr = np.real(self.symbols)
373         ci = np.imag(self.symbols)
374         plt.plot(cr, ci, plot_type)
375         plt.xlabel('Real')
376         plt.ylabel('Imag')
377
378         if self.title is not None:
379             plt.title(self.title)
380
381     def plot_map( self, figure_no = None, disp_x = 0.0, disp_y = 0.0,
382                   rotation = 90, plot_type = 'bo', axis_equal = True):
383
384         self.plot( figure_no = figure_no, plot_type = plot_type)
385
386         for i, bits in enumerate(self.bits_str):
387             symbol = self.symbols[i]
388             bits_str = self.bits_str[i]
389             plt.text( np.real(symbol) + disp_x, np.imag(symbol) + disp_y,
    bits_str,
390                       rotation = rotation)
391
392         if self.title is not None:
393             plt.title(self.title)
394
395         if axis_equal:
396             plt.axis('equal')
397
398     def set_symbols( self, symbols ):
399         self.symbols = symbols
400
401     def set_gray_bits( self, m ):
402         g = gray_code( m )
403         self.m = m
404
405         for i, cw in enumerate(g):
406             self.bits_str.append(cw)
407             self.bits.append( str_to_bitsarray( cw ) )
408             self.map[ cw ] = self.symbols[ i ]
409
410     def bits_to_symbols( self, bits, return_groups = False ):
411         symbols = []
412         bitgroups = []
413         i = 0
414         j = 0
415
416         if not isinstance(bits, str):
417             bits = array_to_str(bits)
418
419         while i < len(bits):
420
421             key = bits[ i : i + self.m ]
422             bitgroups.append( key )
423             symbols.append( self.map[ key ] )
424             i += self.m
```

```
            j += 1

        if not return_groups:
            return np.array(symbols)
        else:
            return np.array(symbols), bitgroups

    def find_closest( self, sample ):
        return np.abs( self.symbols - sample ).argmin()

    def decode( self, sample ):
        i = self.find_closest( sample )
        return [self.symbols[i], self.bits[i], self.bits_str[i] ]
```

Listing 6.2: Η κλάση `constellation`

Στην `constellation.__init__` αρχικοποιούμε την κλάση, θέτοντας κάποιες αρχικές τιμές σε πεδία που θα χρειαστούμε και θα εξηγήσουμε παρακάτω. Το `constellation.symbols` περιέχει τα πλάτη των συμβόλων, το `constellation.bits` τις ακολουθίες από bits που αντιστοιχούν στα σύμβολα, το `constellation.bits_str` περιέχει τις ίδιες ακολουθίες αλλά σε μορφή `string` και το `constellation.title` περιέχει το όνομα του αστερισμού, ενώ το `constellation.m` αποθηκεύει τον αριθμό των bit ανά σύμβολο m. Το `constellation.map` θα χρησιμοποιηθεί για να αποθηκεύει την αντιστοίχιση των συνδυασμών bit σε σύμβολα όπως π.χ. στην εξίσωση 6.9.

Η κλάση `constellation` όπως θα καταλάβατε είναι μία γενική κλάση. Στην `__init__` δεν γίνεται κάποια ανάθεση αρχικών πλατών όπως αυτή που περιγράφεται στην (6.19). Τις ιδιαιτερότητες του M-PAM θα τις υλοποιήσουμε σε μία δεύτερη κλάση `pam_constellation` που θα κληρονομεί μεθόδους από την `constellation`. Όταν θα υλοποιήσουμε κλάσεις που θα περιγράφουν άλλα συστήματα διαμόρφωσης τότε και αυτές θα κληρονομούν μεθόδους της `constellation` που είναι κοινές και επομένως δεν χρειάζεται να τις υλοποιήσουμε για κάθε κλάση. Ας αναφερθούμε συνοπτικά στα γνωρίσματα και τις μεθόδους της κλάσης `constellation`. Σε ότι αφορά τα γνωρίσματα:

- το `symbols` είναι θα περιέχει τα πλάτη των συμβόλων A_k. Για την περίπτωση του $4-PAM$, σύμφωνα και με την εικόνα 6.2, τα στοιχεία του πίνακα θα είναι -3, -1, 1 και 3.
- το `bits` είναι μία *λίστα* που περιέχει ως στοιχεία numpy arrays που περιέχουν τα bits που αντιστοιχούν σε κάθε ένα από τα πλάτη των συμβόλων. Έτσι π.χ. για $M=4$, το `bits` περιέχει τα στοιχεία $(0,0)$, $(0,1)$, $(1,1)$ και $(1,0)$.
- το `bits_str` είναι μία *λίστα* που περιέχει ως στοιχεία `stings` που περιέχουν τα bits που αντιστοιχούν σε κάθε ένα από τα πλάτη των συμβόλων. Έτσι π.χ. για $M=4$, το `bits_str` περιέχει τα strings "00", "01", "11" και "10".
- το `map` είναι ένα λεξικό που θα μας βοηθήσει να αντιστοιχίσουμε συνδυασμούς bit σε σύμβολα. Έτσι για παράδειγμα στην περίπτωση όπου έχουμε 4-PAM με τα πλάτη που περιγράφονται στην εικόνα 6.2, το `map` θα είναι:

$$\begin{aligned} 00 &\rightarrow -3 \\ 01 &\rightarrow -1 \\ 11 &\rightarrow +1 \\ 10 &\rightarrow +3 \end{aligned}$$

Σε ότι αφορά τις μεθόδους εδώ κάνουμε μία σύντομη αναφορά και στις παρακάτω ενότητες θα δούμε περισσότερες λεπτομέρειες για τη φιλοσοφία πίσω από την υλοποίηση τους:

- `avg_power`: Υπολογίζει την μέση ισχύ των συμβόλων.
- `plot`: Κάνει την γραφική παράσταση των πλατών των συμβόλων που υπάρχουν στο γνώρισμα `symbols`.
- `plot`: Κάνει την γραφική παράσταση των παραπάνω συμβόλων αναγράφοντας όμως και την ομάδα bit που αντιστοιχεί στο κάθε ένα.
- `set_symbols`: στην ουσία καταχωρεί κάποιες τιμές στο γνώρισμα `symbols`
- `set_gray_bits`: Εδώ υπολογίζουμε τα bits ενός κώδικα Gray του οποίου το μήκος των λέξεων m το περνάμε ως παράμετρο στην μέθοδο. Αφού κατασκευάσουμε τον κώδικα Gray $\mathbf{G}_m$ τότε τα αποθηκεύουμε διαδοχικά στα γνωρίσματα `bits` ως numpy arrays και `bits_str` ως stings. Επίσης κατασκευάζουμε ανάλογα το λεξικό του γνωρίσματος `map`.
- `bits_to_symbols`: Η μέθοδος αυτή δέχεται ως είσοδο μία σειρά από bit και επιστρέφει τα σύμβολα στα οποία αντιστοιχεί σύμφωνα με την αντιστοίχιση που έχει καθοριστεί στο γνώρισμα `map`.
- `find_closest`: Μας επιστρέφει το δείκτη του συμβόλου του οποίου η τιμή είναι πιο κοντά στην τιμή `sample`. Χρησιμοποιείται για την αποκωδικοποίηση του μηνύματος.
- `decode`: Αποκωδικοποιεί το δείγμα `sample` επιστρέφοντας τόσο το πιο κοντινό σύμβολο όσο και τα αντίστοιχα bits.

6.5 Η κλάση pam_constellation

```
440 class pam_constellation(constellation):
441
442     def __init__(self, M, beta = 1, title = None, SNRbdB = None):
443         super().__init__(title = title)
444
445         self.M = M
446         self.m = np.log2(M).astype(int)
447         self.SNRbdB = SNRbdB
448
449         symbols = np.zeros( M )
450         for i in range( M ):
451             symbols [ i ] = 2 * i - M + 1
452
453         self.set_symbols( symbols )
454         self.set_gray_bits( self.m )
455
456     def ser(self):
457         SNRb = 10 ** ( self.SNRbdB / 10)
458         q = 6 * SNRb * self.m / (self.M ** 2.0 - 1)
459         return 2 * (self.M-1) / self.M * Qfunction ( np.sqrt(q) )
460
461     def ber(self):
462         return self.ser() / self.m
```

Listing 6.3: Η κλάση pam_constellation

Θα υλοποιήσουμε μία κλάση, την `pam_constellation` για να περιγράψουμε πιο συγκεκριμένα την διαμόρφωση PAM. Στο listing 6.3 δείχνουμε πως έχει υλοποιηθεί αυτή η κλάση. Βλέπουμε πως η `pam_constellation` κληρονομεί όλα τα γνωρίσματα και τις μεθόδους της `constellation` και διαφοροποιείται την `__init__` όπου αφού αρχικά καλείται η `__init__` του γονέα (δηλαδή της κλάσης `constellation`) και αρχικοποιούνται τα γνωρίσματα που είδαμε

στην ενότητα 6.4. Στη συνέχεια αρχικοποιούνται διάφορα γνωρίσματα: το `M` και το `m` είναι το πλήθος των συμβόλων και ο αριθμός των bit/σύμβολο αντίστοιχα και διέπονται από την (6.8). Επίσης αρχικοποιούνται και τα σύμβολα στο γνώρισμα `symbols` καθότι γνωρίζουμε ότι δίνονται από την (6.19) και φτιάχνεται η αντιστοίχιση συμβόλων και ομάδων bits με την `set_gray_bits`.

Υπάρχουν και άλλες δύο μέθοδοι της `pam_constellation`, οι `ser` και `ber` που θα συζητήσουμε παρακάτω.

6.6 Μέση ισχύς του σήματος

Ένα πράγμα που σίγουρα μας ενδιαφέρει είναι η αναμενόμενη τιμή $\mathbb{E}\left\{x_k^2\right\}$ των πλατών x_k των συμβόλων στην (6.4). Θυμηθείτε ότι η PSD $S_X(f)$ ενός ψηφιακού σήματος δίνεται από την σχέση (5.147) η οποία στην περίπτωση του σήματος (6.4) γράφεται ως εξής:

$$S_X(f) = \frac{\sigma_x^2}{T_S}|P(f)|^2 \tag{6.20}$$

όπου $\sigma_x^2 = \mathbb{E}\left\{x_k^2\right\}$. Αν θεωρήσουμε ότι τα x_k λαμβάνουν τιμές από τα A_p με την ίδια πιθανότητα τότε θα έχουμε:

$$\Pr\{x_k = A_p\} = \frac{1}{M} \tag{6.21}$$

Για παράδειγμα στην περίπτωση του 4-PAM της εικόνας 6.2 θα έχουμε:

$$\Pr\{x_k = -3\} = \Pr\{x_k = -1\} = \Pr\{x_k = +1\} = \Pr\{x_k = +3\} = \frac{1}{4} \tag{6.22}$$

Η μέση ισχύς των συμβόλων επομένως γράφεται ως εξής:

$$\mathbb{E}\left\{x_k^2\right\} = \sum_{p=0}^{M-1} A_p^2 \Pr\{x_k = A_p\} = \frac{1}{M}\sum_{p=0}^{M-1} A_p^2 \tag{6.23}$$

Παρατηρείστε ότι η (6.23) είναι ακριβώς ο τρόπος που χρησιμοποιεί η μέθοδος `avg_power` της κλάσης `constellation` στο listing 6.2. Στην περίπτωση των πλατών (6.19), θα έχουμε επομένως:

$$\mathbb{E}\left\{x_k^2\right\} = \frac{1}{M}\sum_{p=0}^{M-1} A_p^2 = \frac{\beta^2}{M}\sum_{p=0}^{M-1}(2k - M + 1)^2 = \frac{\beta^2}{M}\sum_{p=0}^{M-1}(4k^2 - 4Mk + 1) \tag{6.24}$$

Το άθροισμα στο δεξιό μέρος της (6.24) γράφεται ως εξής:

$$\sum_{p=0}^{M-1}(4k^2 - 4Mk + 1) = 4\sum_{p=0}^{M-1} k^2 + 4M\sum_{p=0}^{M-1} k + M \tag{6.25}$$

Τα παραπάνω αθροίσματα υπολογίζονται από γνωστούς τύπους:

$$\sum_{q=0}^{Q} q = \frac{Q(Q+1)}{2} \tag{6.26}$$

$$\sum_{q=0}^{Q} q^2 = \frac{Q(Q+1)(2Q+1)}{6} \tag{6.27}$$

```
Power computed from avg_power: 85.0
Power computed from PAM formula: 85.0
```

Εικόνα 6.4: Σύγκριση της (6.28) με την (6.23)

Συνδυάζοντας τις παραπάνω εξισώσεις, μπορούμε να δείξουμε ότι:

$$\sigma_x^2 = \mathbb{E}\left\{x_k^2\right\} = \beta^2 \frac{M^2-1}{3} \tag{6.28}$$

```python
from commlib import pam_constellation
beta = 1.0
M = 16

c = pam_constellation(beta = beta, M = M)
p1 = c.avg_power()
print('Power computed from avg_power:', p1)

p2 = beta ** 2.0 * (M**2 - 1) / 3
print('Power computed from PAM formula:', p2)
```

Listing 6.4: pampower.py

Μπορούμε να συγκρίνουμε την (6.28) με την (6.23). Για τον σκοπό αυτό χρησιμοποιούμε το listing 6.4 και στην εικόνα 6.4 βλέπουμε τη σύγκριση των αποτελεσμάτων για $M = 16$.

Η μέση ισχύς του σήματος δίνεται από την (??) την οποία επαναλαμβάνουμε και εδώ:

$$\mathscr{P}_X = \int_{-\infty}^{+\infty} S_X(f)\mathrm{d}f \tag{6.29}$$

Αντικαθιστώντας την (6.20), έχουμε:

$$\mathscr{P}_X = \frac{\sigma_x^2}{T_\mathrm{S}} \int_{-\infty}^{+\infty} |P(f)|^2 \mathrm{d}f = \frac{\sigma_x^2}{T_\mathrm{S}} \int_{-\infty}^{+\infty} |p(t)|^2 \mathrm{d}t \tag{6.30}$$

Στην τελευταία σχέση χρησιμοποιήσαμε την ταυτότητα του Parseval, (4.58). Από την σχέση προκύπτει ότι η μέση ισχύς $\mathscr{P}_X$ υπολογίζεται ως γινόμενο της μέσης ισχύς των συμβόλων σ_x^2 επί την μέση ισχύ του παλμού, $\frac{1}{T_\mathrm{S}} \int |p(t)|^2 \mathrm{d}t$.

Ειδικά στην περίπτωση όπου τα πλάτη δίνονται από την (6.19), χρησιμοποιούμε την (6.28) για να δείξουμε ότι:

$$\mathscr{P}_X = \beta^2 \frac{M^2-1}{3T_\mathrm{S}} \int_{-\infty}^{+\infty} |p(t)|^2 \mathrm{d}t \tag{6.31}$$

6.7 Η επίδραση του θορύβου

Η κυματομορφή PAM $y(t)$ που λαμβάνουμε στο δέκτη θα αποτελείται από το αρχικό σήμα $x(t)$ που δίνεται από την (6.4) εξασθενημένο κατά $\sqrt{L}$ συν τον θόρυβο $n(t)$,

$$y(t) = \sqrt{L}x(t) + n(t) = \sqrt{L}\sum_{k=0}^{N} x_k p(t - kT_\mathrm{S}) + n(t) \tag{6.32}$$

Το L είναι η εξασθένιση του καναλιού μεταξύ του πομπού και του δέκτη. Θα πρέπει να κάνουμε μερικές παραδοχές για να καταλάβουμε πιο εύκολα την επίδραση του θορύβου, κάποιες από τις οποίες έχουμε ήδη κάνει σιωπηλά. Συγκεκριμένα θεωρούμε ότι:

1. Ο θόρυβος $n(t)$ είναι *λευκός*, δηλαδή ότι η PSD είναι σταθερή στο εύρος ζώνης του σήματος $y(t)$ οπότε έχουμε:

$$S_n(f) = \frac{N_0}{2} \tag{6.33}$$

2. Ο θόρυβος είναι *προσθετικός* δηλαδή προστίθεται απευθείας στο σήμα όπως δείχνει και η (6.32).
3. Ο θόρυβος ακολουθεί *Gaussian* κατανομή οπότε κάθε χρονική στιγμή το $n(t)$ είναι μια κανονική τυχαία μεταβλητή.
4. Το $n(t)$ έχει μέση τιμή μηδέν,

$$\mathbb{E}\{n(t)\} = 0 \tag{6.34}$$

5. Το κανάλι είναι γραμμικό και χρονικά αναλλοίωτο (LTI) και έχει ομαλή συνάρτηση μεταφοράς $H(f) = \sqrt{L}$. Ως εκ τούτου οι παλμοί $p(t)$ είναι ίδιοι τόσο για τον πομπό (σήμα $x(t)$) όσο και για τον δέκτη (σήμα $y(t)$).
6. οι παλμοί $p(t)$ είναι περιορισμένοι μέσα στο διάστημα ενός συμβόλου $[-T_S/2, T_S/2]$. Αυτό σημαίνει ότι οι παλμοί $p(t - kT_S)$ δεν επικαλύπτονται και επομένως για $(k - \frac{1}{2})T_S \leq t < (k + \frac{1}{2})T_S$ θα έχουμε από την (6.4):

$$x(t) = x_k p(t - kT_S) \tag{6.35}$$

   Η ενέργεια του παλμού $p(t)$ συμβολίζεται με $\mathscr{E}_p$ και ισούται με

$$\mathscr{E}_p = \int_{-\frac{T_S}{2}}^{\frac{T_S}{2}} |p(t)|^2 \mathrm{d}t \tag{6.36}$$

   ενώ στο δέκτη το σήμα θα γράφεται:

$$y(t) = x_k \sqrt{L} p(t - kT_S) + n(t) \tag{6.37}$$

7. δεδομένου του σήματος $y(t)$, ο δέκτης μπορεί να κάνει μία επεξεργασία πάνω στο σήμα περνώντας το από ένα σύστημα LTI με κρουστική απόκριση $h(t)$ οπότε υπολογίζει ένα σήμα:

$$z(t) = \int_{-\infty}^{+\infty} h(t - \tau) y(\tau) \mathrm{d}\tau \tag{6.38}$$

   Ο δέκτης δειγματοληπτεί το $z(t)$ στις στιγμές $t = t_k = (k + \frac{1}{2})T_S$ και αποφασίζει για τα σύμβολα βάσει των τιμών των δειγμάτων $z_k = z(kT_S)$.
8. Για την αποκωδικοποίηση των συμβόλων, ο δέκτης υπολογίζει την απόσταση των z_k από όλα τα πιθανά σύμβολα A_m λαμβάνοντας υπόψη και την εξασθένιση του καναλιού L. Οπότε υπολογίζονται οι αποστάσεις:

$$d_m = |x_k - A_m|^2 \tag{6.39}$$

   και ο δέκτης επιλέγει (αποκωδικοποιεί) το σύμβολο A_μ με το μικρότερο d_m, $d_\mu = \min\{d_m\}$. Παρατηρείστε ότι αυτός είναι ακριβώς ο τρόπος με τον οποίο έχουμε υλοποιήσει την μέθοδο `decode`: Η `find_closest` υπολογίζει το δείκτη του συμβόλου στο γνώρισμα `symbols` το οποίο ελαχιστοποιεί το $|x_k - A_m|$ και επομένως και το $|x_k - A_m|^2$.

Θα χρησιμοποιήσουμε τις παραπάνω παραδοχές για να υπολογίσουμε τις επιδόσεις των συστημάτων επικοινωνιών που χρησιμοποιούν PAM αλλά και σε άλλου είδους διαμορφώσεις. Θα ξεκινήσουμε δείχνοντας ποιο είναι το βέλτιστο σύστημα LTI για τον δέκτη, επιλέγοντας την καλύτερη δυνατή $h(t)$ για να ελαχιστοποιήσουμε την επίδραση του θορύβου. Πρώτα όμως πρέπει να ορίσουμε το πηλίκο σήμα-προς-θόρυβο που θα χρησιμοποιήσουμε στην βελτιστοποίηση.

6.8 Ο Βέλτιστος Δέκτης

Θεωρώντας ότι βρισκόμαστε στη διάρκεια του πρώτου συμβόλου ($k = 0$), $-\frac{1}{2}T_S \leq t < \frac{1}{2}T_S$, τότε συνδυάζοντας τις (6.38) και (6.37), έχουμε:

$$z(t) = x_0\sqrt{L}\int_{-\infty}^{+\infty} h(t-\tau)p(\tau)\mathrm{d}\tau + \int_{-\infty}^{+\infty} h(t-\tau)n(\tau)\mathrm{d}\tau \tag{6.40}$$

Επιλέγοντας τη χρονική στιγμή $t = t_0$ όπου ο δέκτης δειγματοληπτεί το σήμα $z(t)$ για να εκτιμήσει το πρώτο σύμβολο ($k = 0$), έχουμε:

$$z_0 = z(t_0) = x_0\sqrt{L}\int_{-\infty}^{+\infty} h(t_0-\tau)p(\tau)\mathrm{d}\tau + \int_{-\infty}^{+\infty} h(t_0-\tau)n(\tau)\mathrm{d}\tau \tag{6.41}$$

Η σχέση (6.41) περιέχει μία συνιστώσα z_s του σήματος που οφείλεται στο αρχικό σύμβολο και μία συνιστώσα z_n που οφείλεται στο θόρυβο. Για να υπολογίσουμε την συνιστώσα του σήματος θέτουμε το θόρυβο $n(t)$ ίσο με μηδέν και έχουμε:

$$z_\mathrm{s} = x_0\sqrt{L}\int_{-\infty}^{+\infty} h(t_0-\tau)p(\tau)\mathrm{d}\tau \tag{6.42}$$

Επίσης αν θέσουμε το πλάτος του συμβόλου x_0 ίσο με ο, θα λάβουμε την συνιστώσα του θορύβου,

$$z_\mathrm{n} = \int_{-\infty}^{+\infty} h(t_0-\tau)n(\tau)\mathrm{d}\tau \tag{6.43}$$

Ο βέλτιστος δέκτης θα πρέπει να ελαχιστοποιεί το z_n και να μεγιστοποιεί το z_s. Η συνιστώσα z_n είναι τυχαία μεταβλητή εξαιτίας του τυχαίου σήματος $n(t)$. Μπορούμε να δείξουμε ότι η μέση τιμή $\mathbb{E}\{z_\mathrm{n}\}$ του z_n είναι μηδέν. Πράγματι,

$$\mathbb{E}\{z_\mathrm{n}\} = \mathbb{E}\left\{\int_{-\infty}^{+\infty} h(t_0-\tau)n(\tau)\mathrm{d}\tau\right\} = \int_{-\infty}^{+\infty}\mathbb{E}\{h(t_0-\tau)n(\tau)\}\,\mathrm{d}\tau = \int_{-\infty}^{+\infty} h(t_0-\tau)\,\mathbb{E}\{n(\tau)\}\,\mathrm{d}\tau = 0 \tag{6.44}$$

Η επίδραση του θορύβου καθορίζεται από την διακύμανση του z_n η οποία δεδομένου ότι η μέση του τιμή είναι μηδέν, καθορίζεται από το $\mathbb{E}\{z_\mathrm{n}^2\}$. Όπως είδαμε και στην ενότητα 5.5, η φασματική πυκνότητα ισχύος $S_z(f)$ στην έξοδο δίνεται από την σχέση:

$$S_z(f) = S_n(f)|H(f)|^2 \tag{6.45}$$

Η ισχύς $\mathbb{E}\{z_n^2\}$ θα δίνεται από το ολοκλήρωμα της $S_z(f)$, δηλαδή:

$$\mathbb{E}\{z_\mathrm{n}^2\} = \int_{-\infty}^{\infty} S_n(f)|H(f)|^2\mathrm{d}f = \frac{N_0}{2}\int_{-\infty}^{\infty}|H(f)|^2\mathrm{d}f = \frac{N_0}{2}\int_{-\infty}^{\infty}|h(\tau)|^2\mathrm{d}\tau \tag{6.46}$$

Ο δέκτης θα προσπαθήσει να μεγιστοποιήσει το πηλίκο

$$\frac{\mathbb{E}\left\{z_s^2\right\}}{\mathbb{E}\left\{z_n^2\right\}} = \frac{\mathbb{E}\left\{x_0^2\right\} L}{N_0} \frac{\left(\int_{-\infty}^{+\infty} h(t_0-\tau)p(\tau)\mathrm{d}\tau\right)^2}{\int_{-\infty}^{\infty} |h(\tau)|^2 \mathrm{d}\tau} \tag{6.47}$$

Για να μεγιστοποιήσουμε την (6.47) θα πρέπει να επιλέξουμε το βέλτιστο $h(t)$. Όταν έχουμε ένα ολοκλήρωμα της μορφής

$$I = \left| \int_{-\infty}^{+\infty} f(\tau)g(\tau)\mathrm{d}\tau \right| \tag{6.48}$$

τότε μπορούμε να δείξουμε ότι αυτό μεγιστοποιείται όταν το $f(\tau)$ είναι ανάλογο του $g(t)$, δηλαδή υπάρχει ένα λ, τέτοιο ώστε

$$f(\tau) = \lambda g(\tau) \tag{6.49}$$

οπότε η βέλτιστη επιλογή για την συνάρτηση μεταφοράς του LTI συστήματος είναι

$$h(t_0-\tau) = \lambda p(\tau) \tag{6.50}$$

ή ισοδύναμα:

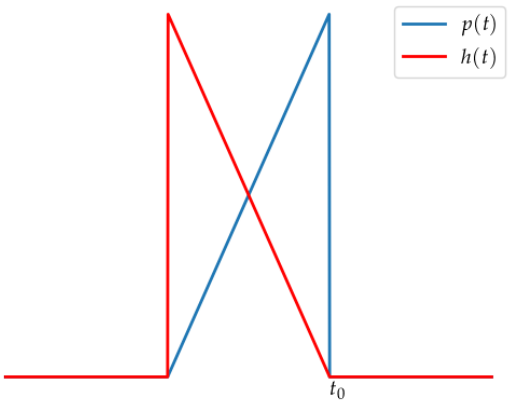
$$h(\tau) = \lambda p(t_0-\tau) \tag{6.51}$$

Από την παραπάνω σχέση προκύπτει ότι ο βέλτιστος δέκτης είναι αυτός του οποίου η απόκριση $h(\tau)$ προκύπτει από την μετατόπιση της βασικής κυματομορφής του παλμού $p(t)$ κατά t_0 (που είναι η χρονική στιγμή της δειγματοληψίας) και την αντιστροφή του ως προς τον άξονα του χρόνου τ. Στην εικόνα 6.5 δείχνουμε ένα παράδειγμα για το πως υπολογίζεται η βέλτιστη απόδοση ενός δέκτη, όταν η βασική κυματομορφή παλμού $p(t)$ είναι τριγωνική και η χρονική στιγμή δειγματοληψίας t_0 είναι στην κορυφή του τριγώνου. Σύμφωνα με την (6.51), θα πρέπει να πάρουμε την συμμετρική συνάρτηση $p(-t)$ του $p(t)$ και να την μετατοπίσουμε στο $t = t_0$ οπότε προκύπτει η κόκκινη συνάρτηση στην εικόνα.

Σε ότι αφορά την σταθερά λ, αυτή μπορούμε να δείξουμε ότι δεν παίζει κανέναν ιδιαίτερο ρόλο στο πηλίκο $\mathbb{E}\left\{z_s^2\right\} / \mathbb{E}\left\{z_s^2\right\}$. Πράγματι αντικαθιστώντας στην (6.47) την (6.51) βλέπουμε ότι:

$$\begin{aligned}\frac{\mathbb{E}\left\{z_s^2\right\}}{\mathbb{E}\left\{z_n^2\right\}} = \frac{\mathbb{E}\left\{x_0^2\right\} L}{N_0} \frac{\left(\int_{-\infty}^{+\infty} h(t_0-\tau)p(\tau)\mathrm{d}\tau\right)^2}{\int_{-\infty}^{\infty} |h(\tau)|^2 \mathrm{d}\tau} = \\ \frac{\mathbb{E}\left\{x_0^2\right\} L}{N_0} \frac{\lambda^2 \left(\int_{-\infty}^{+\infty} p^2(\tau)\mathrm{d}\tau\right)^2}{\lambda^2 \int_{-\infty}^{\infty} |p(\tau)|^2 \mathrm{d}\tau} = \frac{\mathbb{E}\left\{x_0^2\right\} L}{N_0} \frac{\left(\int_{-\infty}^{+\infty} p^2(\tau)\mathrm{d}\tau\right)^2}{\int_{-\infty}^{\infty} |p(\tau)|^2 \mathrm{d}\tau}\end{aligned} \tag{6.52}$$

Επομένως εφόσον το λ απλοποιείται δεν παίζει κάποιο ρόλο στην τιμή του πηλίκο και προκύπτει το παρακάτω συμπέρασμα:



Εικόνα 6.5: Ένα παράδειγμα απόκρισης βέλτιστου δέκτη.

Βέλτιστος Δέκτης

Ο βέλτιστος δέκτης είναι αυτός του οποίου η κρουστική απόκριση συνδέεται με την βασική κυματομορφή βάσει της σχέσης:

$$h(\tau) = p(t_0 - \tau) \tag{6.53}$$

όπου t_0 είναι η χρονική στιγμή δειγματοληψίας της εξόδου του δέκτη.

Εάν θεωρήσουμε ότι ο παλμός $p(t)$ είναι πραγματικός τότε $p^2(t) = |p(t)|^2$ οπότε:

$$\frac{\mathbb{E}\left\{z_s^2\right\}}{\mathbb{E}\left\{z_n^2\right\}} = \frac{\mathbb{E}\left\{x_0^2\right\} L}{N_0} \int_{-\infty}^{+\infty} p^2(\tau) \mathrm{d}\tau = \frac{L \mathbb{E}\left\{x_0^2\right\} \mathscr{E}_p}{N_0} \tag{6.54}$$

που μας δείχνει την τιμή του βέλτιστου $\mathbb{E}\left\{z_s^2\right\} / \mathbb{E}\left\{z_n^2\right\}$

6.9 Το πηλίκο σήμα προς θόρυβο

Η (6.54) εκφράζει το πηλίκο του μέσου τετραγωνικού πλάτους της συνιστώσας του σήματος z_s προς το μέσο τετραγωνικό πλάτος της συνιστώσας του θορύβου, z_n. Είναι προφανές ότι ένα τέτοιου είδους πηλίκο θα είναι ένα μέτρο για το πόσο καλές είναι οι επιδόσεις του συστήματος, εφόσον επιθυμούμε η συνιστώσα z_s να είναι όσο το δυνατόν μεγαλύτερη σε σχέση με το z_n. Ωστόσο το συγκεκριμένο πηλίκο είναι στην *έξοδο* του δέκτη ενώ εμείς θα προτιμούσαμε να έχουμε έναν τρόπο να καταλαβαίνουμε τι συμβαίνει στην *είσοδο* του δέκτη δηλαδή αναφορικά με το σήμα $y(t)$ στην (6.37): πόσο πιο ισχυρό είναι το σήμα που φθάνει στον δέκτη σε σχέση με τον θόρυβο;

Επίσης προσέξτε ότι το $\mathbb{E}\left\{z_s^2\right\}$ είναι το πλάτος του σήματος το οποίο είναι ανάλογο με το πλάτος $\mathbb{E}\left\{x_0^2\right\}$ του *συμβόλου*. Θα προτιμούσαμε να μετράμε κάτι που σχετίζεται με τα *bits*, π.χ. *πως συγκρίνεται η ενέργεια που λαμβάνουμε και αντιστοιχεί σε ένα bit σε σχέση με την ενέργεια του θορύβου;*

Σύμφωνα με την (6.37), η συνιστώσα του σήματος y_s όταν αναφερόμαστε στο πρώτο σύμβολο $k=0$ και προκύπτει $n(t)=0$, ισούται με:

$$y_s(t) = x_0\sqrt{L}p(t) \tag{6.55}$$

Επομένως η ενέργεια του σήματος μέσα στην διάρκεια του συμβόλου θα είναι

$$E_{\mathrm{S}} = \mathbb{E}\left\{\int_{-\frac{T_S}{2}}^{\frac{T_S}{2}} |y_s(t)|^2 \mathrm{d}t\right\} = L\mathbb{E}\left\{x_0^2\right\}\int_{-\frac{T_S}{2}}^{\frac{T_S}{2}} |p(t)|^2 \mathrm{d}t = L\mathbb{E}\left\{x_0^2\right\}\mathscr{E}_p \tag{6.56}$$

Η μέση ισχύς στην είσοδο είναι το πηλίκο της ενέργειας S προς την διάρκεια του συμβόλου S, δηλαδή:

$$P_{\mathrm{S}} = \frac{E_{\mathrm{S}}}{T_{\mathrm{S}}} = \frac{L\mathbb{E}\left\{x_0^2\right\}\mathscr{E}_p}{T_{\mathrm{S}}} \tag{6.57}$$

Σε ότι αφορά την συνιστώσα του θορύβου $n(t)$ θέλει μία προσοχή επειδή ο θόρυβος μπορεί να έχει και συνιστώσες που θα μπορούσαν να απομακρυνθούν από τον δέκτη (π.χ. βρίσκονται εκτός του φάσματος του σήματος $y_s(t)$. Είδαμε στην ενότητα 6.8, ότι στην καλύτερη περίπτωση ο δέκτης μπορεί να περάσει το σήμα και τον θόρυβο από ένα φίλτρο $h(t) = p(t_0 - t)$ οπότε σύμφωνα με την (6.42) θα επιβιώσουν οι συνιστώσες που "ταιριάζουν" με το $h(t)$,

$$z_n = \int_{-\infty}^{+\infty} h(t_0-\tau)n(\tau)\mathrm{d}\tau = \int_{-\infty}^{+\infty} p(\tau)n(\tau)\mathrm{d}\tau \tag{6.58}$$

Οπότε οι "επιζήμιες" συνιστώσες είναι αυτές που επικαλύπτονται με το $p(t)$. Όπως και πριν μπορούμε να γράψουμε

$$\mathbb{E}\left\{z_n^2\right\} = \frac{N_0}{2}\int_{-\infty}^{+\infty} |h(t)|^2 \mathrm{d}t = \frac{N_0}{2}\int_{-\infty}^{+\infty} |p(t)|^2 \mathrm{d}t = \frac{N_0}{2}\mathscr{E}_p \tag{6.59}$$

Η συνιστώσα του σήματος και αυτή θα πολλαπλασιαστεί με το $p(t)$ σύμφωνα με την (6.40). Οπότε ακολουθώντας και την (6.42), θα έχουμε:

$$z_s = x_0\sqrt{L}\int_{-\infty}^{+\infty} |p(t)|^2 \mathrm{d}t = x_0\sqrt{L}\mathscr{E}_p \tag{6.60}$$

και

$$\mathbb{E}\left\{z_s^2\right\} = L\mathbb{E}\left\{x_0^2\right\}\mathscr{E}_p^2 \tag{6.61}$$

Οπότε ένα σήμα στην είσοδο με ισχύ P_{S} μετασχηματίζεται σε μία μεταβλητή z_n που έχει ισχύ όπως αυτή καθορίζεται από την (6.61) όπου:

$$\frac{\mathbb{E}\left\{z_s^2\right\}}{P_{\mathrm{S}}} = \frac{\mathscr{E}_p}{T_{\mathrm{S}}} \tag{6.62}$$

Σε αναλογία με την (6.62), θα μπορούσαμε να θεωρήσουμε μία ισοδύναμη ισχύ θορύβου στην είσοδο P_{N} η οποία έχει ως αποτέλεσμα την εμφάνιση της ισχύος θορύβου $\mathbb{E}\left\{z_n^2\right\}$ και θα καθορίζεται ανάλογα:

$$\frac{\mathbb{E}\left\{z_n^2\right\}}{P_{\mathrm{N}}} = \frac{\mathscr{E}_p}{T_{\mathrm{S}}} \tag{6.63}$$