Assignment № 1

Steven Seppala 2 Feb 2015

Problem 2.9

$$U(\theta,\phi) = \begin{cases} 1 & 0° \leq \theta < 20° \\ .342\csc(\theta) & 20° \leq \theta < 60° \\ 0 & 60° \leq \theta \leq 180° \end{cases}$$

$$D_o = \frac{U_{max}(\theta,\phi)}{P_{rad}}$$

$$P_{rad} = \oiint U(\theta,\phi)d\Omega$$

$$\implies \int_0^{2\pi}\int_0^{\pi} U(\theta,\phi) \implies \int_0^{2\pi}[\int_0^{\pi/9} \sin(\theta)d\theta + \int_{\pi/9}^{\pi/3} .342\csc(\theta)\sin(\theta)d\theta + \int_{\pi/3}^{\pi} 0\sin(\theta)d\theta]$$

$$\implies ([-\cos(\theta]\Big|_{\theta=0}^{\theta=\frac{\pi}{9}} + .342[\theta]\Big|_{\theta=\frac{\pi}{9}}^{\theta=\frac{\pi}{3}}) * 2\pi$$

$$\implies P_{rad} = 1.879$$

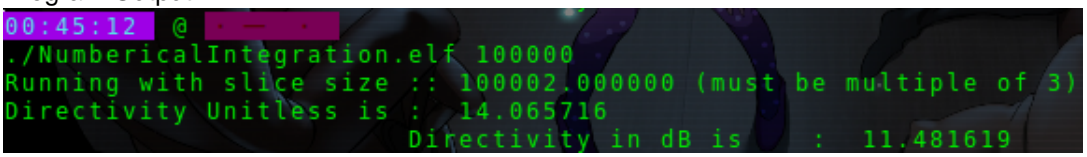
$$\implies D_o = \frac{4_{max}}{1.879} = \frac{4\pi}{1.879} \implies$$

$$\mathsf{D}_o = 6.687 \text{ no units}$$

$$D_{dBi} = 10log_{10}D_o \implies$$

$$\mathsf{D}_{dBi} = 8.253 \text{ dB}$$

Problem 2.26

1. C Program source code is in appendix.
   Program Output ::
   

2. Krauss Approxamation = $\frac{4\pi}{\Theta_{1r}\Theta_{2r}}$
   $\sqrt{\Theta}_{1r} = \sqrt{\Theta}_{2r} := U(\theta) = .5$
   $.5 = [\frac{\sin(\pi \sin(\theta))}{\pi \sin(\theta)}]^2 \implies \theta = .458$
   $\frac{4\pi}{.916^2} = 14.9768$ dimensionless
   $D_{dB} = 10log_{10}(14.9768) = 11.75$ dB

3. Tai-Peeira Approxamation =
   $\frac{22.181}{2(.916)^2} = 13.217$ dimensionless
   $D_{dB} = 10log_{10}(13.217) = 11.21$ dB

Problem 2.35

$$E_w^i = (\hat{a}x + j\hat{a}y)E_o e^{+jkz}$$
$$E_a = (\hat{a} + 2\hat{a}y)E_1 \frac{e^{-jkr}}{r}$$

- $\Delta_\phi = \phi_x - \phi_y = (2n+1)\frac{\pi}{2}$
  Circularly polarized because the y component is 90°out of phase with the x component due to its j amplitude. This will make the phase delta of x and y always be odd multiples of $\frac{\pi}{2}$ as well as E_x being equal to E_y.
- The rotation is clockwise because of the y component has a higher amplitude than the x component and will pull the rotation clockwise.
- $E_w^i = \frac{\hat{a}_x + j\hat{a}_y}{\sqrt{5}}\sqrt{5}E_1\frac{e^{-jkz}}{z}$
  $\implies$ 2 components with 0° phase difference $\implies$ Linear polarization
- Since it is linear it has no rotation.
- $\hat{\rho}_w = \frac{\hat{a}_x + j\hat{a}_y}{\sqrt{2}}, \hat{\rho}_a = \frac{\hat{a}_x + 2\hat{a}_y}{\sqrt{5}}$
  PLF = $|\hat{\rho}_w \cdot \hat{\rho}_a|^2 \implies \frac{|1+j|^2}{10} = \frac{5}{10}$
  PLF = .5 dimensionless
  $PLF_{dB} = 10log_{10}(.5) = -3$ dB

Problem 2.40

$\hat{\rho_a} = \frac{4\hat{a}_x + j\hat{a}_y}{\sqrt{17}}$
$\tau = 45°$
$PLF = |\frac{4\hat{a}_x + j\hat{a}_y}{\sqrt{17}} * \frac{\hat{a}_x + j\hat{a}_y}{\sqrt{2}}|^2$
$\implies \frac{|4+j|^2}{34} \implies .5$ dimensionless
$10log_{10}(.5) = -3.0103$ dB

Problem 2.70

1. Gain = Efficiency * Directivity
   $\varepsilon = \frac{A_{em}}{A_p} \implies$
   $e_o * \varepsilon = \frac{A_{em}}{A_p} * e_o \implies$
   $1 = \frac{\frac{\lambda^2}{4\pi} D_o}{10cm^2} e_o \implies$
   $10cm^2 = \frac{9cm^2}{4\pi} Gain \implies Gain = \frac{40\pi}{9}$
   $\implies 13.9626$ dimentionless
   $10log_{10}(13.9626) = 11.45 \implies$ Gain $= 11.45$ dB

2. $P_T = A_e * W_i$
   $P_T = 10\frac{(mW)}{(cm)^2}(10cm^2)(.5) \implies P_T = 50$ mW

Problem 2.80

$$U(\theta, \phi) = \begin{cases} \cos^4(\theta) & 0° \leq \theta < 90° \\ 0 & 90° \leq \theta \leq 180° \end{cases}$$

$$\begin{cases} 1 & 0° \leq \phi \leq 360° \end{cases}$$

$A_{em} = \frac{\lambda^2}{4\pi} D_o$

$P_{rad} = \int_0^{2\pi} \int_0^{2\pi} \cos^4(\theta)\sin(\theta) d\theta d\phi$

$\rightarrow$ calculator $\rightarrow$

$2\pi[.2] \implies P_{rad} = 1.256$

$D_o = 4\pi \frac{}{1.256 = 10.0051}$

$\implies \frac{9cm^2}{4\pi}(10.0051) = 7.165 * 10^{-2}$ m^2

Problem 2.86

$P_r = e_r * D_r * \frac{\lambda^2}{4\pi} * \frac{P_t D_t}{4\pi R^2}$

$\implies P_r = 20db \frac{10w * 20dB}{4\pi(50)^2\lambda^2} \frac{\lambda^2}{4\pi}$

$\implies \frac{400dB * 10w}{16\pi^2 2500} = 10$mW

APPENDIX

```
1
2  /*************************************************************************
3      This program performs a numerical integration of
4      [ (sin( pi sin(theta) ) / (pi sin(theta)) ] ^2
5      between 0 and PI. It then uses the result to find the
6      directivity of an antenna with the U(theta, phi)
7      characteristics.
8
9      Operations are therded to speed up calculations
10     as a signficant number of slices must be chosen to
11     obtain accuracy. An example is 1000 for fair accurasy.
12
13    Copyright (C) 2015 Steven Seppala
14
15 This program is free software: you can redistribute it and/or modify
16 it under the terms of the GNU General Public License as published by
17 the Free Software Foundation, either version 3 of the License, or
18 (at your option) any later version.
19
20 This program is distributed in the hope that it will be useful,
21 but WITHOUT ANY WARRANTY; without even the implied warranty of
22 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
23 GNU General Public License for more details.
24
25 You should have received a copy of the GNU General Public License
26 along with this program.  If not, see <http://www.gnu.org/licenses/>.
27 *************************************************************************/
28 #include <stdio.h>
29 #include <math.h>
30 #include <pthread.h>
31 #include <unistd.h>
32 #include <time.h>
33 #include <signal.h>
34 #include <stdlib.h>
35 #include <sys/wait.h>
36
37
38 #define PI 3.14159265358979323846264338327950288419716939937510582
39
40 float h_val, x_i, a_bound, b_bound, n_slices;
41 long double temp_vals[4];
42 int first, second = 0;
43
44 void * threes (void *t)
45 {
46     long double a_n;
47     long double seq_threes;
48     long double x_i;
```

```
49      long tid = (long)t;
50      long double threes_sum = 0;
51      for(int i = 1; i < n_slices; i++)
52      {
53          a_n = .25*((6*i)-(pow((-1),i))-3);
54          if (fmodl(a_n, i) == 0)
55          {
56              x_i = a_n * (PI / n_slices);
57          //  printf("Value of i :: %d  Value of x_i :: %Lf \n", i, x_i);
58              seq_threes = 3*(pow(((sin(PI*sin(x_i)))/(PI*sin(x_i))),2))*sin(x_i);
59              threes_sum = threes_sum + seq_threes;
60          }
61      }
62
63      temp_vals[0] = threes_sum;
64      pthread_exit((void*) t);
65      return EXIT_SUCCESS;
66  }
67
68  void * twos (void *t)
69  {
70      long double a_n;
71      long double seq_twos;
72      long double x_i;
73      long double twos_sum = 0;
74      long tid = (long)t;
75      for(int i = 3; i < n_slices; i++)
76      {
77          a_n = 3*i;
78          if (fmodl(a_n, i) == 0)
79          {
80              x_i = a_n * (PI / n_slices);
81          //  printf("Value of i :: %d  Value of x_i :: %Lf \n", i, x_i);
82              seq_twos = 2*(pow(((sin(PI*sin(x_i)))/(PI*sin(x_i))),2))*sin(x_i);
83              twos_sum = twos_sum + seq_twos;
84          }
85      }
86
87      temp_vals[1] = twos_sum;
88      pthread_exit((void*) t);
89      return EXIT_SUCCESS;
90  }
91
92  void* ends(void *t)
93  {
94      long double a_n, a_0;
95      long double x_n = n_slices;
96      long double x_0 = 0;
97      long tid = (long)t;
98      pid_t end_to_end;
99      //printf("End to ends:: x_n ==%Lf \n",x_n);
```

```
100
101     end_to_end = fork();
102     if (end_to_end == 0)
103     {
104         x_0 = 0 * (PI / n_slices);
105         a_0 = 1;
106     //  printf("a_0 :: %Lf :: x_0 :: %Lf\n",a_0, x_0);
107         exit(1);
108     }
109     else
110     {
111         x_n = n_slices * (PI / n_slices);
112         a_n = pow(((sin(PI*sin(x_n)))/(PI*sin(x_n))),2)*sin(x_i);
113     //  printf("a_n :: %Lf || x_n :: %Lf\n",a_n, x_n);
114         wait(NULL);
115     }
116
117     temp_vals[2] = a_0 + a_n;
118     pthread_exit((void*) t);
119     return EXIT_SUCCESS;
120 }
121
122
123 int main(int argc, char *argv[])
124 {
125
126     pid_t id;
127     pthread_t ends_thread, twos_and_threes[3];
128     pthread_mutex_t thread_locker;
129     void * status;
130     long t;
131
132     n_slices =(atof(argv[1])) ;
133     n_slices = ((n_slices + 3 - 1) / 3) * 3 ;
134
135     printf("Running with slice size :: %lf (must be multiple of 3)\n", n_slices);
136     pthread_attr_t attr;
137     pthread_attr_init(&attr);
138     pthread_mutex_init(&thread_locker, NULL);
139     pthread_attr_setdetachstate(&attr, PTHREAD_CREATE_JOINABLE);
140     //Child Process
141     if(pthread_create(&twos_and_threes[0], &attr,ends,(void*) t))
142         return EXIT_FAILURE;
143     if(pthread_create(&twos_and_threes[1],&attr,twos,(void*)t))
144         return EXIT_FAILURE;
145     if(pthread_create(&twos_and_threes[2],&attr,threes,(void*)t))
146         return EXIT_FAILURE;
147
148     pthread_attr_destroy(&attr);
149     for(int i = 0 ; i < 3 ; i++)
150         if(pthread_join(twos_and_threes[i], &status))
```

```
151         {
152             fprintf(stderr, "Catastrophic Failure");
153             return EXIT_FAILURE;
154         }
155     /*
156     twos();
157     threes();
158     ends();
159     */
160 
161     temp_vals[3] = temp_vals[0]+temp_vals[1] +temp_vals[2];
162     //printf( "Temp vals :: %Lf :: %Lf :: %Lf :: %Lf\n",\
163             temp_vals[0],temp_vals[1],temp_vals[2],temp_vals[3]);
164 
165     long double h_val = ((PI-0) / n_slices);
166     long double result = temp_vals[3] * (3*h_val) * (.125) ;
167     //printf("Result == %Lf \n",result);
168 
169     long double P_rad = result*2*PI;
170     long double directivity = (2*PI*1) / (P_rad);
171 
172     printf("Directivity Unitless is :  %Lf \n \
173             Directivity in dB is    :  %f \n", directivity, (10*log10f(directivity↩
                 )));
174 
175 
176     pthread_exit(NULL);
177 }
```

*Makefile for compiling the above code

```
#
#  Copyright 2014 Steven T Seppala <steven.t.seppala@gmail.com>
#
#  This program is free software; you can redistribute it and/or modify
#  it under the terms of the GNU General Public License as published by
#  the Free Software Foundation; either version 2 of the License, or
#  (at your option) any later version.
#
#  This program is distributed in the hope that it will be useful,
#  but WITHOUT ANY WARRANTY; without even the implied warranty of
#  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
#  GNU General Public License for more details.
#
#  You should have received a copy of the GNU General Public License
#  along with this program; if not, write to the Free Software
#  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston,
#  MA 02110-1301, USA.
#
#


CC= gcc
CC2= clang
LD= ld -r
CFLAGS= -std=gnu99 -g -O3 -lm -pthread
FL2=
FL3=
TGTS=
OBJ=
RM= /bin/rm -f

all : NumbericalIntegration

NumbericalIntegration:
    $(CC2) $(CFLAGS) $@.c -o $@.elf
clean :
    $(RM) *.elf *.txt
```