Solutions for HW1

Problem 1.

1. for $p = 1$ the set is the solution of the equation $|x + y| = 1$ It forms a diamond centered at 0 with the lines connecting the points $(1, 0)$, $(0, 1)$, $(0, -1)$, $(-1, 0)$.

2. for $p = 2$ the set is a circle centered at 0.

3. for $p = \infty$ we have either one (or both) coordinates is equal to zero which gives us a square with vertices at the points $(1, 1)$, $(-1, -1)$, $(1, -1)$, $(-1, 1)$.

For the second part of the problem we have $\|z\|_2 = 1 \Rightarrow x^2 = 1 - y^2$. Substituting the result into $\|z\|_1$ we may find the local maximum of the function $f(z) = f(x, y) = |x+y| = |y \pm \sqrt{1 - y^2}|$ We may remove the absolute value here as the answer will be symmetric and (for simplicity) consider the first quadrant only. Setting the derivative to 0 we obtain:

$$1 - \frac{1}{2}\frac{1}{\sqrt{1 - y^2}}(-2y) = 0.$$

$$y^2 = 1 - y^2 \Rightarrow y = \pm\frac{1}{\sqrt{2}}$$

Substituting y to find x and then both into the norm yields the result.

Note. The intuitive understanding of this result could be seen if we combine the sets for $p = 1$ and $p = 2$ on the same graph. In this case the target point will lie on the circle and will be the most distant from the "diamond".

Problem 2. None of those functions are norms as they fail the triangle inequality. For $< 0p < 1$ the function F_p is concave so it could be seen on the graph (or checked by direct substitution, like $x = (1, 0)$, $y = (0, 1)$).

Problem 3. Here is the example of the program written in pseudocode:

init=0; target="; *initializing the variables, target is a string here*
read(init); *get the initial decimal number*
repeat
digit=init mod 2 *the remainder after division by 2*
target=target+digit *add the obtained digit to our representation*
init=init div 2 *proceed to the next order of* 2^k
until init<2 *so we are done reducing it*
target=target+init *adding the last one (init is less than 2 now so it could be seen as binary digit)*

Note. To get the actual binary number we need to "flip" the representation obtained in this way as the order of digits here is lowest power-highest power.

Problem 4.

Hint: to make the orthogonalization process a bit simpler we may first find *orthogonal* basis and then normalize vectors by dividing them by their norms.

Now

$$f_1 = e_1 = (0, 2, -2, 0)$$

$$f_2 = e_2 - \frac{(f_1, e_2)}{(f_1, f_1)} f_1 = (0,1,0,-1) - \frac{1}{4}(0,2,-2,0) = (0, \frac{1}{2}, \frac{1}{2}, -1)$$

$$f_3 = e_3 - \frac{(f_1, e_3)}{(f_1, f_1)} f_1 - \frac{(f_2, e_3)}{(f_2, f_2)} f_2 =$$

$$= (0,-1,1,-1) - (-1)(0,2,-2,0) - \frac{2}{3}(0, \frac{1}{2}, \frac{1}{2}, -1) = (0, \frac{2}{3}, -\frac{4}{3}, -\frac{1}{3})$$

And after division by the corresponding norms we obtained the required result.

Problem 5. Though (theoretically) matlab's random could create a matrix with all rows and columns linearly dependent, most probably the result would be something like that:

$$\begin{pmatrix} 1 & 0 & 0 & ... & a & d \\ ... & & & ... & & \\ 0 & 1 & 0 & ... & b & e \\ 0 & 0 & 1 & ... & c & f \end{pmatrix}$$

From which it is clearly seen that though rows could be linearly independent form a linear subspace, columns could not (as there are 8 6-dimensional rows).