Part 1 Machine precision

Computer uses binary number to represent numbers. Use this information to write a short Matlab code to obtain the machine precision as accurate as possible.

Solution:
The following is the algorithm that was used to compute machine precision in Matlab:

Algorithm 1: Machine precision

```
function epsilon = machine_precision()

    epsilon = 1.0;
    while 1.0 + (epsilon/2) ~= 1.0
        epsilon = (epsilon / 2);

end
```

This function starts with a double (epsilon), and cuts it in half until the computer cannot distinguish between 1.0 and the sum of 1.0 and epsilon (the loop termination condition). Running this function in the Matlab console, we observe the following:

```
>> machine_precision()
    ans = 2.2204e-16
```

Comparing this output to the built-in Matlab function eps(), which "returns the distance from 1.0 to the next larger double-precision number"[1].

```
>> eps()
    ans = 2.2204e-16
```

Thus, machine precision for a double on my laptop (a 64-bit machine) is 2.2204×10^{-16}.

Part 2 Relative error of power series

Expand

$$f(x) = \frac{\sin x}{x} - 1$$

in a power series about $x = 0$. Calculate the number of terms which are necessary to ensure a relative error of 10^{-7} and of 10^{-16} for any $x \in [0, 1]$.

Solution:
The following is the algorithm that was used to compute the relative error of a power series representation of a given function:

Algorithm 2: Relative error of a power series

```
function o = power_error(f, x, desired_error)

    o = 0;
    error = realmax;

    while error > desired_error
        o = o + 1;
        T = taylor(f, "Order", o);
        error = relative_error(double(subs(f,x)), double(subs(T,x)));
    end
end
```

Algorithm 3: Relative error

```
function error = relative_error(a, b)
    error = abs((a-b)/a);
end
```

This algorithm computes a taylor polynomial approximating a given function f with different numbers of terms in the polynomial until the desired error is obtained, at which point the algorithm outputs the number of terms required to obtain the desired error.

Running this algorithm in a Matlab console using $f = \frac{\sin x}{x} - 1$, $x = 0.5$, and a desired error of 10^{-7} and 10^{-16}, we observe the following:

```
>> power_error(f, 0.5, 10e-16)
    ans = 15

>> power_error(f, 0.5, 10e-7)
    ans = 7
```

Not surprisingly, more terms in the taylor polynomial are requred in order to get a smaller error. Now we will test values on the extreme ends of the domain $[0, 1]$.

```
>> power_error(f, 0.99999, 10e-16)
    ans = 17

>> power_error(f, 0.99999, 10e-7)
    ans = 9

>> power_error(f, 0.00001, 10e-16)
    ans = 5

>> power_error(f, 0.00001, 10e-7)
    ans = 3
```

Part 3 Relative error of k-digit rounding

Given a real number on a computer with k-digit rounding arithmetic, analytically estimate the relative error bound for the computer representation.

Solution:
Define a function in a computer programming language $round(x) = \hat{x}$ that rounds a real number x using k-digit rounding arithmetic. For this problem we consider real numbers in the format $0.d_1d_2d_3...d_n$. We consider the case where the $k+1$ digit in the real number x is 5 or greater:

$$\left|\frac{x-\hat{x}}{x}\right| = \left|\frac{0.d_1...d_kd_{\mathrm{k+1}}...d_n \times 10^n - (0.d_1...d_k \times 10^n + 10^{\mathrm{n-k}})}{0.d_1...d_kd_{.}..d_n \times 10^n}\right|$$

$$= \left|\frac{0.d_1...d_kd_{\mathrm{k+1}}...d_n \times 10^n - (0.d_1...d_k + 10^{-\mathrm{k}}) \times 10^n}{0.d_1...d_kd_{.}..d_n \times 10^n}\right|$$

$$= \left|\frac{0.d_1...d_kd_{\mathrm{k+1}}...d_n \times 10^n - 0.d_1...d_kr \times 10^n}{0.d_1...d_kd_{.}..d_n \times 10^n}\right|$$

Where r is d_{k+1} in the real number rounded up to the next digit.

$$= \left|\frac{0.d_1...d_kd_{\mathrm{k+1}}...d_n - 0.d_1...d_kr}{0.d_1...d_kd_{.}..d_n}\right|$$

We know that $0.0...r > 0.0...d_kd_{\mathrm{k+1}}...d_n$ since it was rounded up.

$$\leq \frac{0.0...5}{0.d_1...d_kd_{.}..d_n} = \frac{0.5}{0.d_1...d_kd_{.}..d_n} \times 10^{-\mathrm{k}} = \frac{0.5}{0.1} \times 10^{-\mathrm{k}}$$

Therefore:

$$\left|\frac{x-\hat{x}}{x}\right| = 0.5 \times 10^{-\mathrm{k}+1}$$

Part 4 Sources

1: https://www.mathworks.com/help/matlab/ref/eps.html