UNIVERSIDAD DE COSTA RICA

Aritmética de precisión finita

Mario De León Urbina

Escuela de Matemática

13 de agosto de 2023

Contenidos

1 Contenidos

UNIVERSIDAD DE COSTA RICA

1 Introducción

1 Introducción

En el colegio a menudo utilizamos calculadoras científicas para realizar cálculos numéricos y algebraicos. Pero, ¿se ha preguntado usted cómo es que funciona su calculadora, al menos matemáticamente hablando?

1 Introducción

En el colegio a menudo utilizamos calculadoras científicas para realizar cálculos numéricos y algebraicos. Pero, ¿se ha preguntado usted cómo es que funciona su calculadora, al menos matemáticamente hablando? Usemos la CASIO *fx*-570ES PLUS para mostrar lo siguiente.

1 Introducción

En el colegio a menudo utilizamos calculadoras científicas para realizar cálculos numéricos y algebraicos. Pero, ¿se ha preguntado usted cómo es que funciona su calculadora, al menos matemáticamente hablando? Usemos la CASIO *fx*-570ES PLUS para mostrar lo siguiente.

1 Introducción

En el colegio a menudo utilizamos calculadoras científicas para realizar cálculos numéricos y algebraicos. Pero, ¿se ha preguntado usted cómo es que funciona su calculadora, al menos matemáticamente hablando? Usemos la CASIO *fx*-570ES PLUS para mostrar lo siguiente.

- Si ponemos 10^{99} el output que aparece es 1×10^{99}, pero si ponemos 10^{100} aparece el mensaje `Math ERROR`. ¿Será que la calculadora no puede almacenar números de esta magnitud "tan grande"?

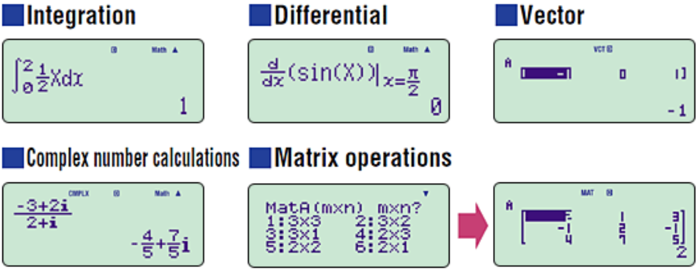
1 Introducción

En el colegio a menudo utilizamos calculadoras científicas para realizar cálculos numéricos y algebraicos. Pero, ¿se ha preguntado usted cómo es que funciona su calculadora, al menos matemáticamente hablando? Usemos la CASIO *fx*-570ES PLUS para mostrar lo siguiente.

- Si ponemos 10^{99} el output que aparece es 1×10^{99}, pero si ponemos 10^{100} aparece el mensaje `Math ERROR`. ¿Será que la calculadora no puede almacenar números de esta magnitud "tan grande"?
- Ahora, si ponemos 10^{-99} el output que aparece es 1×10^{-99}, pero si ponemos 10^{-100} el resultado es igual a 0. ¿Será que la calculadora no puede almacenar números de esta magnitud "tan pequeña"?

UNIVERSIDAD DE COSTA RICA

Figura: Pantalla de una calculadora CASIO *fx*-570ES.



1 Introducción

1 Introducción

UNIVERSIDAD DE COSTA RICA

- Consideremos la ecuación $x^2 + 100\,000x + 1 = 0$. Si usamos el `Mode`$\to 5 \to 3$ obtenemos que $x_1 = -1 \times 10^{-5}$, $x_2 = -99999.99999$. Las soluciones exactas, calculadas por métodos conocidos son

- Consideremos la ecuación $x^2 + 100\,000x + 1 = 0$. Si usamos el `Mode`$\rightarrow 5 \rightarrow 3$ obtenemos que $x_1 = -1 \times 10^{-5}$, $x_2 = -99999.99999$. Las soluciones exactas, calculadas por métodos conocidos son

$$
\begin{aligned}
x_1 &= \sqrt{2\,499\,999\,999} - 50\,000 \\
&\approx -0.000010000000001000000000200000000005\ldots
\end{aligned}
$$

- Consideremos la ecuación $x^2 + 100\,000x + 1 = 0$. Si usamos el `Mode`$\rightarrow 5 \rightarrow 3$ obtenemos que $x_1 = -1 \times 10^{-5}$, $x_2 = -99999.99999$. Las soluciones exactas, calculadas por métodos conocidos son

$$\begin{aligned} x_1 &= \sqrt{2\,499\,999\,999} - 50\,000 \\ &\approx -0.000010000000001000000000200000000005\ldots \\ x_2 &= -\sqrt{2\,499\,999\,999} - 50\,000 \\ &\approx -99999.99998999999999899999999980000\ldots \end{aligned}$$

- Consideremos la ecuación $x^2 + 100\,000x + 1 = 0$. Si usamos el `Mode`$\rightarrow 5 \rightarrow 3$ obtenemos que $x_1 = -1 \times 10^{-5}$, $x_2 = -99999.99999$. Las soluciones exactas, calculadas por métodos conocidos son

$$\begin{aligned} x_1 &= \sqrt{2\,499\,999\,999} - 50\,000 \\ &\approx -0.0000100000000001000000000200000000005\ldots \\ x_2 &= -\sqrt{2\,499\,999\,999} - 50\,000 \\ &\approx -99999.9999899999999998999999999980000\ldots \end{aligned}$$

Es notable que al tener una cantidad determinada de dígitos en el output las representaciones de números irracionales en la calculadora “eliminan” dígitos de la representación exacta.

2 Contenidos

2 Representación punto flotante

2 Representación punto flotante

Una computadora tiene únicamente un número finito de bits. Esto requiere representaciones aproximadas de los números reales.

2 Representación punto flotante

Una computadora tiene únicamente un número finito de bits. Esto requiere representaciones aproximadas de los números reales.

Definition (Representación punto flotante)

2 Representación punto flotante

Una computadora tiene únicamente un número finito de bits. Esto requiere representaciones aproximadas de los números reales.

Definition (Representación punto flotante)

Sea $\beta \geq 2$ una base (usualmente par), t la precisión (número de dígitos en base β). Un número **punto flotante** se representa en esta base como

2 Representación punto flotante

Una computadora tiene únicamente un número finito de bits. Esto requiere representaciones aproximadas de los números reales.

Definition (Representación punto flotante)

Sea $\beta \geq 2$ una base (usualmente par), t la precisión (número de dígitos en base β). Un número **punto flotante** se representa en esta base como

$$\xi = \pm(d_1.d_2...d_t)_\beta \times \beta^e = \pm m \times \beta^e$$

UNIVERSIDAD DE COSTA RICA

2 Representación punto flotante

Una computadora tiene únicamente un número finito de bits. Esto requiere representaciones aproximadas de los números reales.

Definition (Representación punto flotante)

Sea $\beta \geq 2$ una base (usualmente par), t la precisión (número de dígitos en base β). Un número **punto flotante** se representa en esta base como

$$\xi = \pm(d_1.d_2...d_t)_\beta \times \beta^e = \pm m \times \beta^e$$

en donde $d_k \in \{0, 1, ..., \beta - 1\}$ son los **dígitos significativos,** m **mantisa** y tiene t dígitos; $e \in \mathbb{Z}$ es el **exponente**.

2 Representación punto flotante

Una computadora tiene únicamente un número finito de bits. Esto requiere representaciones aproximadas de los números reales.

Definition (Representación punto flotante)

Sea $\beta \geq 2$ una base (usualmente par), t la precisión (número de dígitos en base β). Un número **punto flotante** se representa en esta base como

$$\xi = \pm(d_1.d_2...d_t)_\beta \times \beta^e = \pm m \times \beta^e$$

en donde $d_k \in \{0, 1, ..., \beta - 1\}$ son los **dígitos significativos,** m **mantisa** y tiene t dígitos; $e \in \mathbb{Z}$ es el **exponente**.

- Si $d_1 \neq 0$ el número se dice **normalizado**. Un número punto flotante normalizado tal que $d_1 = 0$ implica entonces que $d_2 = d_3 = ... = d_t = 0$.

2 Representación punto flotante

Una computadora tiene únicamente un número finito de bits. Esto requiere representaciones aproximadas de los números reales.

Definition (Representación punto flotante)

Sea $\beta \geq 2$ una base (usualmente par), t la precisión (número de dígitos en base β). Un número **punto flotante** se representa en esta base como

$$\xi = \pm(d_1.d_2...d_t)_\beta \times \beta^e = \pm m \times \beta^e$$

en donde $d_k \in \{0, 1, ..., \beta - 1\}$ son los **dígitos significativos,** m **mantisa** y tiene t dígitos; $e \in \mathbb{Z}$ es el **exponente**.

- Si $d_1 \neq 0$ el número se dice **normalizado**. Un número punto flotante normalizado tal que $d_1 = 0$ implica entonces que $d_2 = d_3 = ... = d_t = 0$.
- En caso contrario el número se llama **subnormal**.

2 Representación punto flotante

2 Representación punto flotante

Observaciones:

Observaciones:

- Note que dicho número es igual a

$$\pm(d_1 + d_2\beta^{-1} + d_3\beta^{-2} + \ldots + d_t\beta^{-(t-1)}) \times \beta^e, \quad 0 \leq d_i < \beta$$

Observaciones:

- Note que dicho número es igual a

$$\pm(d_1 + d_2\beta^{-1} + d_3\beta^{-2} + ... + d_t\beta^{-(t-1)}) \times \beta^e, \quad 0 \leq d_i < \beta$$

- Si el exponente de dos números punto flotante es el mismo, se dice que tienen la misma *magnitud*. Los exponentes máximo y mínimo se denotan como $e_{\max}$ y $e_{\min}$, respectivamente, y se tiene que (usualmente) $e_{\min} < 0 < e_{\max}$.

Observaciones:

- Note que dicho número es igual a

$$\pm(d_1 + d_2\beta^{-1} + d_3\beta^{-2} + ... + d_t\beta^{-(t-1)}) \times \beta^e, \quad 0 \leq d_i < \beta$$

- Si el exponente de dos números punto flotante es el mismo, se dice que tienen la misma *magnitud*. Los exponentes máximo y mínimo se denotan como $e_{\max}$ y $e_{\min}$, respectivamente, y se tiene que (usualmente) $e_{\min} < 0 < e_{\max}$.
  - Entonces hay $e_{\max} - e_{\min} + 1$ posibles exponentes.

Observaciones:

- Note que dicho número es igual a

$$\pm(d_1 + d_2\beta^{-1} + d_3\beta^{-2} + \ldots + d_t\beta^{-(t-1)}) \times \beta^e, \quad 0 \leq d_i < \beta$$

- Si el exponente de dos números punto flotante es el mismo, se dice que tienen la misma *magnitud*. Los exponentes máximo y mínimo se denotan como e_{max} y e_{min}, respectivamente, y se tiene que (usualmente) $e_{\text{min}} < 0 < e_{\text{max}}$.
  - Entonces hay $e_{\text{max}} - e_{\text{min}} + 1$ posibles exponentes.
- De manera alternativa podemos representar ξ de manera única en la forma

Observaciones:

- Note que dicho número es igual a

$$\pm(d_1 + d_2\beta^{-1} + d_3\beta^{-2} + \ldots + d_t\beta^{-(t-1)}) \times \beta^e, \quad 0 \leq d_i < \beta$$

- Si el exponente de dos números punto flotante es el mismo, se dice que tienen la misma *magnitud*. Los exponentes máximo y mínimo se denotan como $e_{\max}$ y $e_{\min}$, respectivamente, y se tiene que (usualmente) $e_{\min} < 0 < e_{\max}$.
  - Entonces hay $e_{\max} - e_{\min} + 1$ posibles exponentes.
- De manera alternativa podemos representar ξ de manera única en la forma

$$\xi = \pm m \times \beta^{e+1-t}, \qquad m \in \{\beta^{t-1}, \beta^{t-1}+1, \ldots, \beta^t - 1\} \subset \mathbb{N}, \quad e \in \mathbb{Z}$$

2 Representación punto flotante

[1] O *machine numbers set*.

2 Representación punto flotante

Definition (Conjunto de números reales de precisión finita)

[1] O *machine numbers set*.

Definition (Conjunto de números reales de precisión finita)

Sean β, $t \in \mathbb{N}$, $0 \leq d_i < \beta$ para $i > 1$, $d_1 \neq 0$ y $L = e_{\text{min}}$, $U = e_{\text{max}}$. Se define *el conjunto de números reales de precisión finita*[1] como

[1] O *machine numbers set*.

Definition (Conjunto de números reales de precisión finita)

Sean β, $t \in \mathbb{N}$, $0 \leq d_i < \beta$ para $i > 1$, $d_1 \neq 0$ y $L = e_{\text{min}}$, $U = e_{\text{max}}$. Se define *el conjunto de números reales de precisión finita*[1] como

$$\mathbb{F}(\beta, t, L, U) := \{0\} \cup \left\{ \xi \in \mathbb{R}^* : \xi = \pm \beta^e \times \sum_{i=1}^{t} d_i \beta^{1-i} \right\}$$

[1] O *machine numbers set*.

Definition (Conjunto de números reales de precisión finita)

Sean β, $t \in \mathbb{N}$, $0 \leq d_i < \beta$ para $i > 1$, $d_1 \neq 0$ y $L = e_{\text{min}}$, $U = e_{\text{max}}$. Se define *el conjunto de números reales de precisión finita*[1] como

$$\mathbb{F}(\beta, t, L, U) := \{0\} \cup \left\{ \xi \in \mathbb{R}^* : \xi = \pm\beta^e \times \sum_{i=1}^{t} d_i \beta^{1-i} \right\}$$

- Si no hay ambigüedad, el conjunto se denota como $\mathbb{F}_{\beta,t}$.

[1] O *machine numbers set*.

2 Representación punto flotante

Theorem (Cardinalidad de $\mathbb{F}_{\beta,t}$)

La cantidad de elementos de $\mathbb{F}(\beta, t, L, U)$, *cuando* $|L|, |U| < \infty$, *es exactamente*

2 Representación punto flotante

Theorem (Cardinalidad de $\mathbb{F}_{\beta,t}$)

La cantidad de elementos de $\mathbb{F}(\beta, t, L, U)$*, cuando* $|L|, |U| < \infty$*, es exactamente*

$$\mathit{card}\ \mathbb{F}(\beta, t, L, U) = 2(\beta - 1)\beta^{t-1}(U - L + 1) + 1$$

Theorem (Cardinalidad de $\mathbb{F}_{\beta,t}$)

La cantidad de elementos de $\mathbb{F}(\beta, t, L, U)$, cuando $|L|, |U| < \infty$, es exactamente

$$card\ \mathbb{F}(\beta, t, L, U) = 2(\beta - 1)\beta^{t-1}(U - L + 1) + 1$$

Observación:

También se puede probar que

Theorem (Cardinalidad de $\mathbb{F}_{\beta,t}$)

La cantidad de elementos de $\mathbb{F}(\beta, t, L, U)$, cuando $|L|, |U| < \infty$, es exactamente

$$\mathit{card}\ \mathbb{F}(\beta, t, L, U) = 2(\beta - 1)\beta^{t-1}(U - L + 1) + 1$$

Observación:

También se puede probar que

- $x_{\text{mín}} = \beta^L$

Theorem (Cardinalidad de $\mathbb{F}_{\beta,t}$)

La cantidad de elementos de $\mathbb{F}(\beta, t, L, U)$, cuando $|L|, |U| < \infty$, es exactamente

$$\mathit{card}\ \mathbb{F}(\beta, t, L, U) = 2(\beta - 1)\beta^{t-1}(U - L + 1) + 1$$

Observación:

También se puede probar que

- $x_{\text{mín}} = \beta^L$
- $x_{\text{máx}} = (1 - \beta^{-t})\beta^{U+1}$

Observación:

Hay dos razones por las cuales un número real x no puede ser representado exactamente como un número punto flotante.

Observación:

Hay dos razones por las cuales un número real x no puede ser representado exactamente como un número punto flotante.

1. La primera se ilustra con el número en base diez $\mathbf{0.1}$, el cual tiene una representación decimal finita, pero en binario su representación es $(0.0\overline{0011})_2$, luego dicho número en binario se encuentra estrictamente entre dos números punto flotante y a la vez no es ninguno de ellos.

2 Representación punto flotante

Observación:

Hay dos razones por las cuales un número real x no puede ser representado exactamente como un número punto flotante.

1. La primera se ilustra con el número en base diez **0.1**, el cual tiene una representación decimal finita, pero en binario su representación es $(0.0\overline{0011})_2$, luego dicho número en binario se encuentra estrictamente entre dos números punto flotante y a la vez no es ninguno de ellos.

2. La segunda razón es que el número $x \in \mathbb{R}$ podría ser muy grande/pequeño en valor absoluto y se escape del rango. A esto se le conoce como **overflow** (sobredesbordamiento) o **underflow** (subdesbordamiento) respectivamente.

Observación:

Hay dos razones por las cuales un número real x no puede ser representado exactamente como un número punto flotante.

1. La primera se ilustra con el número en base diez **0.1**, el cual tiene una representación decimal finita, pero en binario su representación es $(0.0\overline{0011})_2$, luego dicho número en binario se encuentra estrictamente entre dos números punto flotante y a la vez no es ninguno de ellos.

2. La segunda razón es que el número $x \in \mathbb{R}$ podría ser muy grande/pequeño en valor absoluto y se escape del rango. A esto se le conoce como **overflow** (sobredesbordamiento) o **underflow** (subdesbordamiento) respectivamente.
   - Por defecto, las computadoras "atrapan" esos casos limítrofes asignando los valores $\pm\infty$ y cero sin previo aviso.

2 Normas IEEE 754

[2] Institute of Electrical and Electronics Engineers

2 Normas IEEE 754

La IEEE[2] 754 reconoce los siguientes símbolos:

- $\pm\infty$, i.e. como el valor de $\pm 1/0$;
- `NaN` *not a number*, i.e. como el resultado de $0/0$ o $\infty - \infty$.

[2] Institute of Electrical and Electronics Engineers

La IEEE[2] 754 reconoce los siguientes símbolos:

- $\pm\infty$, i.e. como el valor de $\pm 1/0$;
- `NaN` *not a number*, i.e. como el resultado de $0/0$ o $\infty - \infty$.

Esencialmente cada computadora desde 1985 implementa el estandard IEEE 754, el cual ofrece dos formatos binarios para la *hardware arithmethic* (por defecto `MATLAB` usa doble precisión, pero también permite el uso de precisión simple). La idea es representar un número asignando los bits de la siguiente manera:

[2]Institute of Electrical and Electronics Engineers

2 Normas IEEE 754

La IEEE[2] 754 reconoce los siguientes símbolos:

- $\pm\infty$, i.e. como el valor de $\pm 1/0$;
- `NaN` *not a number*, i.e. como el resultado de $0/0$ o $\infty - \infty$.

Esencialmente cada computadora desde 1985 implementa el estandard IEEE 754, el cual ofrece dos formatos binarios para la *hardware arithmethic* (por defecto `MATLAB` usa doble precisión, pero también permite el uso de precisión simple). La idea es representar un número asignando los bits de la siguiente manera:

signo	exponente	mantisa

[2] Institute of Electrical and Electronics Engineers

2 Precisión Simple

2 Precisión Simple

- Utiliza 32 bits (4 bytes) distribuidos así: 1 bit para el signo, 8 bits para el exponente y 23 bits para la mantisa. Tiene una precisión de 24 dígitos binarios. El exponente toma valores en $[-126, 127]$ con sesgo 127.

2 Precisión Simple

- Utiliza 32 bits (4 bytes) distribuidos así: 1 bit para el signo, 8 bits para el exponente y 23 bits para la mantisa. Tiene una precisión de 24 dígitos binarios. El exponente toma valores en $[-126, 127]$ con sesgo 127.

$\pm$	$a_1 a_2 a_3 \ldots a_8$	$b_1 b_2 b_3 \ldots b_{23}$

2 Precisión Simple

- Utiliza 32 bits (4 bytes) distribuidos así: 1 bit para el signo, 8 bits para el exponente y 23 bits para la mantisa. Tiene una precisión de 24 dígitos binarios. El exponente toma valores en $[-126, 127]$ con sesgo 127.

$\pm$	$a_1 a_2 a_3 \ldots a_8$	$b_1 b_2 b_3 \ldots b_{23}$

- El valor de e que se guarda en los bits $a_1, a_2, ..., a_8$ corresponde a $e + 127$, y el épsilon máquina es $\varepsilon_m = 2^{-23}$.

2 Precisión Simple

2 Precisión Simple

Ejemplo:

Convirtamos $(32995)_{10}$ a su representación de precisión simple:

Ejemplo:

Convirtamos $(32995)_{10}$ a su representación de precisión simple:

haciendo las respectivas operaciones se obtiene que

$$m = (32995)_{10} = (1000000011100011)_2$$

Ejemplo:

Convirtamos $(32995)_{10}$ a su representación de precisión simple:

haciendo las respectivas operaciones se obtiene que

$$m = (32995)_{10} = (1000000011100011)_2$$

Normalizando, $m = (1.000000011100011)_2 \times 2^{15} \longrightarrow e = 15.$

Ejemplo:

Convirtamos $(32995)_{10}$ a su representación de precisión simple:

haciendo las respectivas operaciones se obtiene que

$$m = (32995)_{10} = (1000000011100011)_2$$

Normalizando, $m = (1.000000011100011)_2 \times 2^{15} \longrightarrow e = 15.$

El sesgo en precisión simple es

$$127 \Rightarrow e + 127 = 15 + 127 = 142 = (10001110)_2$$

2 Precisión Simple

Ejemplo:

Convirtamos $(32995)_{10}$ a su representación de precisión simple:

haciendo las respectivas operaciones se obtiene que

$$m = (32995)_{10} = (1000000011100011)_2$$

Normalizando, $m = (1.000000011100011)_2 \times 2^{15} \longrightarrow e = 15.$

El sesgo en precisión simple es

$$127 \Rightarrow e + 127 = 15 + 127 = 142 = (10001110)_2$$

Entonces el número se almacena en precisión simple como

$$(32995)_{10} = 0\,10001110\,00000001110001100000000$$

2 Precisión Doble

2 Precisión Doble

- Utiliza 64 bits (8 bytes) distribuidos así: 1 bit para el signo, 11 bits para el exponente y 52 bits para la mantisa. Tiene una precisión de 53 dígitos binarios. El exponente toma valores en $[-1022, 1023]$ con sesgo 1023.

2 Precisión Doble

- Utiliza 64 bits (8 bytes) distribuidos así: 1 bit para el signo, 11 bits para el exponente y 52 bits para la mantisa. Tiene una precisión de 53 dígitos binarios. El exponente toma valores en $[-1022, 1023]$ con sesgo 1023.

$\pm$	$a_1 a_2 a_3 \ldots a_{11}$	$b_1 b_2 b_3 \ldots b_{52}$

2 Precisión Doble

- Utiliza 64 bits (8 bytes) distribuidos así: 1 bit para el signo, 11 bits para el exponente y 52 bits para la mantisa. Tiene una precisión de 53 dígitos binarios. El exponente toma valores en $[-1022, 1023]$ con sesgo 1023.

$\pm$	$a_1 a_2 a_3 \ldots a_{11}$	$b_1 b_2 b_3 \ldots b_{52}$

- El valor de e que se guarda en los bits $a_1, a_2, a_3, \ldots, a_{11}$ corresponde a e^{1023}. Además, la precisión es $p = 53$ y el épsilon máquina es $\varepsilon_m = 2^{-52}$

Ejemplo:

Convirtamos $(32995)_{10}$ a su representación de precisión doble:

Ejemplo:

Convirtamos $(32995)_{10}$ a su representación de precisión doble:

haciendo las respectivas operaciones se obtiene que

$$m = (32995)_{10} = (1000000011100011)_2$$

Normalizando, $m = (1.000000011100011)_2 \times 2^{15} \longrightarrow e = 15$.

El sesgo en precisión doble es

$$1023 \Rightarrow e + 1023 = 15 + 1023 = 1038 = (10000001110)_2$$

Ejemplo:

Convirtamos $(32995)_{10}$ a su representación de precisión doble:

haciendo las respectivas operaciones se obtiene que

$$m = (32995)_{10} = (1000000011100011)_2$$

Normalizando, $m = (1.000000011100011)_2 \times 2^{15} \longrightarrow e = 15.$

El sesgo en precisión doble es

$$1023 \Rightarrow e + 1023 = 15 + 1023 = 1038 = (10000001110)_2$$

Entonces el número se almacena en precisión doble como

$$(32995)_{10} = 0\,10000001110\,000000011100011 0\ldots 0$$

3 Representación flotante

3 Representación flotante

Supongamos que se está utilizando el sistema de precisión doble de la IEEE. Sea $\mathbb{F}$ como el conjunto de números en punto flotante del sistema de precisión doble.

Supongamos que se está utilizando el sistema de precisión doble de la IEEE. Sea $\mathbb{F}$ como el conjunto de números en punto flotante del sistema de precisión doble.

Definition (Función `fl`)

Considere la función $\mathtt{fl} : \mathbb{R} \to \mathbb{F}$, donde a cada número real ξ se le asigna el correspondiente número en punto flotante $\mathtt{fl}(\xi)$, utilizando redondeo al valor más cercano. Es decir, si $\xi_- \leqslant \xi < \xi_+$, donde $\xi_-, \xi_+ \in \mathbb{F}$ son dos números en punto flotante consecutivos, se define

Supongamos que se está utilizando el sistema de precisión doble de la IEEE. Sea $\mathbb{F}$ como el conjunto de números en punto flotante del sistema de precisión doble.

Definition (Función `fl`)

Considere la función $\texttt{fl} : \mathbb{R} \to \mathbb{F}$, donde a cada número real ξ se le asigna el correspondiente número en punto flotante $\texttt{fl}(\xi)$, utilizando redondeo al valor más cercano. Es decir, si $\xi_- \leqslant \xi < \xi_+$, donde $\xi_-, \xi_+ \in \mathbb{F}$ son dos números en punto flotante consecutivos, se define

$$\texttt{fl}(\xi) = \begin{cases} \xi_-, & \text{si } |\xi - \xi_-| \leqslant |\xi - \xi_+| \\ \xi_+, & \text{en caso contrario} \end{cases}$$

3 Épsilon máquina

3 Épsilon máquina

Definition

El número ε_m más pequeño tal que

3 Épsilon máquina

Definition

El número ε_m más pequeño tal que

$$\texttt{fl}(1+\varepsilon_m) > 1$$

se denomina **épsilon máquina** o **macheps**.

3 Épsilon máquina

Definition

El número ε_m más pequeño tal que

$$\mathtt{fl}(1 + \varepsilon_m) > 1$$

se denomina **épsilon máquina** o **macheps**.

Nota:

El épsilon máquina es la diferencia entre 1 y el número siguiente $x > 1$ con $x \in \mathbb{F}$ que se puede almacenar de forma exacta.

3 Épsilon máquina

Definition

El número ε_m más pequeño tal que

$$\mathtt{fl}(1+\varepsilon_m) > 1$$

se denomina **épsilon máquina** o **macheps**.

Nota:

El épsilon máquina es la diferencia entre 1 y el número siguiente $x > 1$ con $x \in \mathbb{F}$ que se puede almacenar de forma exacta.

- Épsilon máquina para precisión simple es $\varepsilon_m = 2^{-23} \approx 1.19 \times 10^{-7}$.

3 Épsilon máquina

Definition

El número ε_m más pequeño tal que

$$\texttt{fl}(1+\varepsilon_m) > 1$$

se denomina **épsilon máquina** o **macheps**.

Nota:

El épsilon máquina es la diferencia entre 1 y el número siguiente $x > 1$ con $x \in \mathbb{F}$ que se puede almacenar de forma exacta.

- Épsilon máquina para precisión simple es $\varepsilon_m = 2^{-23} \approx 1.19 \times 10^{-7}$.
- Para precisión doble es $\varepsilon_m = 2^{-52} \approx 2.22 \times 10^{-16}$.

3 Errores

3 Errores

Theorem

Sea $x \in \mathbb{R}$ un número que puede ser representado de manera normal en un sistema de punto flotante con precisión t. Entonces, existe $\delta \in \mathbb{R}$ tal que

Theorem

Sea $x \in \mathbb{R}$ un número que puede ser representado de manera normal en un sistema de punto flotante con precisión t. Entonces, existe $\delta \in \mathbb{R}$ tal que

$$fl(x) = x(1+\delta)$$

donde $|\delta| < \varepsilon_m/2$

Theorem

Sea $x \in \mathbb{R}$ un número que puede ser representado de manera normal en un sistema de punto flotante con precisión t. Entonces, existe $\delta \in \mathbb{R}$ tal que

$$fl(x) = x(1+\delta)$$

donde $|\delta| < \varepsilon_m/2$

Definition

Theorem

Sea $x \in \mathbb{R}$ un número que puede ser representado de manera normal en un sistema de punto flotante con precisión t. Entonces, existe $\delta \in \mathbb{R}$ tal que

$$fl(x) = x(1 + \delta)$$

donde $|\delta| < \varepsilon_m/2$

Definition

Sea $x^* = \mathtt{fl}(x)$ Se define el *error absoluto* ε como

$$x^* = x + \varepsilon,$$

Theorem

Sea $x \in \mathbb{R}$ un número que puede ser representado de manera normal en un sistema de punto flotante con precisión t. Entonces, existe $\delta \in \mathbb{R}$ tal que

$$fl(x) = x(1+\delta)$$

donde $|\delta| < \varepsilon_m/2$

Definition

Sea $x^* = \mathtt{fl}(x)$ Se define el *error absoluto* ε como

$$x^* = x + \varepsilon,$$

y el *error relativo* δ como

$$x^* = x(1+\delta)$$

3 Errores

3 Errores

En la aritmética punto flotante, el error relativo es muy apropiado porque cada número es representado con una precisión relativa similar.

3 Errores

En la aritmética punto flotante, el error relativo es muy apropiado porque cada número es representado con una precisión relativa similar. Cuando $x = 0$ o x es muy cercano a cero, es mejor considerar el error absoluto.

En la aritmética punto flotante, el error relativo es muy apropiado porque cada número es representado con una precisión relativa similar. Cuando $x = 0$ o x es muy cercano a cero, es mejor considerar el error absoluto.

- Los errores al representar x por $\mathrm{fl}(x)$ también suelen representarse así:

3 Errores

En la aritmética punto flotante, el error relativo es muy apropiado porque cada número es representado con una precisión relativa similar. Cuando $x = 0$ o x es muy cercano a cero, es mejor considerar el error absoluto.

- Los errores al representar x por $\mathrm{fl}(x)$ también suelen representarse así:

Error absoluto

$$\varepsilon = |x - \mathtt{fl}(x)|$$

En la aritmética punto flotante, el error relativo es muy apropiado porque cada número es representado con una precisión relativa similar. Cuando $x = 0$ o x es muy cercano a cero, es mejor considerar el error absoluto.

- Los errores al representar x por $\text{fl}(x)$ también suelen representarse así:

Error absoluto

$$\varepsilon = |x - \texttt{fl}(x)|$$

Error relativo

$$\delta = \frac{|x - \texttt{fl}(x)|}{|x|}, \quad x \neq 0$$

3 Dígitos significativos

3 Dígitos significativos

Para una base β se tiene que el error relativo, al aproximar x por algún valor, con precisión t, está acotado de la siguiente manera:

3 Dígitos significativos

Para una base β se tiene que el error relativo, al aproximar x por algún valor, con precisión t, está acotado de la siguiente manera:

$$|\delta| = \left|\frac{\varepsilon}{x}\right| < \frac{1/2 \times \beta^{e-t+1}}{1 \times \beta^{e}} = 1/2 \times \beta^{-t+1}$$

Para una base β se tiene que el error relativo, al aproximar x por algún valor, con precisión t, está acotado de la siguiente manera:

$$|\delta| = \left|\frac{\varepsilon}{x}\right| < \frac{1/2 \times \beta^{e-t+1}}{1 \times \beta^{e}} = 1/2 \times \beta^{-t+1}$$

En el caso de $\beta = 10$, base 10, se tiene la siguiente definición:

Para una base β se tiene que el error relativo, al aproximar x por algún valor, con precisión t, está acotado de la siguiente manera:

$$|\delta| = \left|\frac{\varepsilon}{x}\right| < \frac{1/2 \times \beta^{e-t+1}}{1 \times \beta^{e}} = 1/2 \times \beta^{-t+1}$$

En el caso de $\beta = 10$, base 10, se tiene la siguiente definición:

Definition

Se dice que $\mathrm{fl}(x)$ aproxima a x con p dígitos significativos si p es el mayor entero no negativo para cual se cumple que

UNIVERSIDAD DE COSTA RICA

3 Dígitos significativos

Para una base β se tiene que el error relativo, al aproximar x por algún valor, con precisión t, está acotado de la siguiente manera:

$$|\delta| = \left|\frac{\varepsilon}{x}\right| < \frac{1/2 \times \beta^{e-t+1}}{1 \times \beta^{e}} = 1/2 \times \beta^{-t+1}$$

En el caso de $\beta = 10$, base 10, se tiene la siguiente definición:

Definition

Se dice que $\mathrm{fl}(x)$ aproxima a x con p dígitos significativos si p es el mayor entero no negativo para cual se cumple que

$$\delta < \frac{1}{2} \times 10^{1-p} = 5 \times 10^{-p}$$

3 Truncamiento y Redondeo

Definition (Truncamiento)

Sea $x = \pm(d_1.d_2...d_t d_{t+1}...)_\beta \times \beta^e$ entonces podemos aproximar x por *truncamiento* hasta el dígito t como $\mathtt{fl}_T(x) = \pm(d_1.d_2...d_t)_\beta \times \beta^e$.

3 Truncamiento y Redondeo

Definition (Truncamiento)

Sea $x = \pm(d_1.d_2...d_t d_{t+1}...)_\beta \times \beta^e$ entonces podemos aproximar x por *truncamiento* hasta el dígito t como $\mathtt{fl}_T(x) = \pm(d_1.d_2...d_t)_\beta \times \beta^e$.

Definition (Redondeo)

Sea $x = \pm(d_1.d_2...d_t d_{t+1}...)_\beta \times \beta^e$ entonces podemos aproximar x por *redondeo* hasta el dígito t como

UNIVERSIDAD DE COSTA RICA

3 Truncamiento y Redondeo

Definition (Truncamiento)

Sea $x = \pm(d_1.d_2 \ldots d_t d_{t+1} \ldots)_\beta \times \beta^e$ entonces podemos aproximar x por *truncamiento* hasta el dígito t como $\texttt{fl}_T(x) = \pm(d_1.d_2 \ldots d_t)_\beta \times \beta^e$.

Definition (Redondeo)

Sea $x = \pm(d_1.d_2 \ldots d_t d_{t+1} \ldots)_\beta \times \beta^e$ entonces podemos aproximar x por *redondeo* hasta el dígito t como

$$\texttt{fl}_R(x) = \begin{cases} \pm(d_1.d_2 \ldots d_t)_\beta \times \beta^e & \text{si } d_{t+1} < \frac{\beta}{2}, \\ \pm(d_1.d_2 \ldots (d_t + 1))_\beta \times \beta^e & \text{si } d_{t+1} \geq \frac{\beta}{2} \end{cases}$$

3 Truncamiento y Redondeo

3 Truncamiento y Redondeo

La operación de redondeo $\texttt{fl} : \mathbb{R} \to \mathbb{F}$ mapea $\xi \in \mathbb{R}$ al número máquina más cercano $\texttt{fl}(\xi) = \hat{\xi} \in \mathbb{F}$. Dicha operación posee las siguientes propiedades:

3 Truncamiento y Redondeo

La operación de redondeo $\texttt{fl} : \mathbb{R} \to \mathbb{F}$ mapea $\xi \in \mathbb{R}$ al número máquina más cercano $\texttt{fl}(\xi) = \hat{\xi} \in \mathbb{F}$. Dicha operación posee las siguientes propiedades:

- Monotonía: $\xi \leq \eta \Rightarrow \texttt{fl}(\xi) \leq \texttt{fl}(\eta)$;

UNIVERSIDAD DE COSTA RICA

La operación de redondeo $\mathtt{fl} : \mathbb{R} \to \mathbb{F}$ mapea $\xi \in \mathbb{R}$ al número máquina más cercano $\mathtt{fl}(\xi) = \hat{\xi} \in \mathbb{F}$. Dicha operación posee las siguientes propiedades:

- Monotonía: $\xi \leq \eta \Rightarrow \mathtt{fl}(\xi) \leq \mathtt{fl}(\eta)$;
- Idempotencia: $\mathtt{fl}(\xi) = \xi$ para $\xi \in \mathbb{F}$.

3 Truncamiento y Redondeo

La operación de redondeo $\texttt{fl} : \mathbb{R} \to \mathbb{F}$ mapea $\xi \in \mathbb{R}$ al número máquina más cercano $\texttt{fl}(\xi) = \hat{\xi} \in \mathbb{F}$. Dicha operación posee las siguientes propiedades:

- Monotonía: $\xi \leq \eta \Rightarrow \texttt{fl}(\xi) \leq \texttt{fl}(\eta)$;
- Idempotencia: $\texttt{fl}(\xi) = \xi$ para $\xi \in \mathbb{F}$.

Theorem (Cota superior para el error relativo)

3 Truncamiento y Redondeo

La operación de redondeo $\mathtt{fl} : \mathbb{R} \to \mathbb{F}$ mapea $\xi \in \mathbb{R}$ al número máquina más cercano $\mathtt{fl}(\xi) = \hat{\xi} \in \mathbb{F}$. Dicha operación posee las siguientes propiedades:

- Monotonía: $\xi \leq \eta \Rightarrow \mathtt{fl}(\xi) \leq \mathtt{fl}(\eta)$;
- Idempotencia: $\mathtt{fl}(\xi) = \xi$ para $\xi \in \mathbb{F}$.

Theorem (Cota superior para el error relativo)

Sea $x \neq 0$ real en el rango y $fl(x)$ su representación punto flotante. Entonces se cumple que

La operación de redondeo $\mathtt{fl} : \mathbb{R} \to \mathbb{F}$ mapea $\xi \in \mathbb{R}$ al número máquina más cercano $\mathtt{fl}(\xi) = \hat{\xi} \in \mathbb{F}$. Dicha operación posee las siguientes propiedades:

- Monotonía: $\xi \leq \eta \Rightarrow \mathtt{fl}(\xi) \leq \mathtt{fl}(\eta)$;
- Idempotencia: $\mathtt{fl}(\xi) = \xi$ para $\xi \in \mathbb{F}$.

Theorem (Cota superior para el error relativo)

Sea $x \neq 0$ real en el rango y $fl(x)$ su representación punto flotante. Entonces se cumple que

$$\frac{|x - fl(x)|}{|x|} \leq \mu = \begin{cases} \beta^{1-t} & \textit{(truncamiento)} \\ \frac{1}{2}\beta^{1-t} & \textit{(redondeo)} \end{cases}$$

4 Contenidos

UNIVERSIDAD DE COSTA RICA

4 Operaciones punto flotante

4 Operaciones punto flotante

La IEEE 754 define que las operaciones aritméticas y la operación raíz cuadrada son calculadas para números máquina por un redondeo correcto para el resultado exacto que se persigue: si se denota la realización de una operación como $\star$ por $\hat{\star}$, entonces para $x, y \in \mathbb{F}$ se tiene que

4 Operaciones punto flotante

La IEEE 754 define que las operaciones aritméticas y la operación raíz cuadrada son calculadas para números máquina por un redondeo correcto para el resultado exacto que se persigue: si se denota la realización de una operación como $\star$ por $\hat{\star}$, entonces para $x, y \in \mathbb{F}$ se tiene que

$$x \hat{\star} y = \texttt{fl}(x \star y), \qquad \star \in \{+, -, \cdot, \div, \sqrt{\ }\}$$

4 Operaciones punto flotante

La IEEE 754 define que las operaciones aritméticas y la operación raíz cuadrada son calculadas para números máquina por un redondeo correcto para el resultado exacto que se persigue: si se denota la realización de una operación como $\star$ por $\hat{\star}$, entonces para $x, y \in \mathbb{F}$ se tiene que

$$x \hat{\star} y = \texttt{fl}(x \star y), \qquad \star \in \{+, -, \cdot, \div, \sqrt{\ }\}$$

Esto así, se tiene el modelo estándar para los números máquina y sus operaciones:

4 Operaciones punto flotante

La IEEE 754 define que las operaciones aritméticas y la operación raíz cuadrada son calculadas para números máquina por un redondeo correcto para el resultado exacto que se persigue: si se denota la realización de una operación como $\star$ por $\hat{\star}$, entonces para $x, y \in \mathbb{F}$ se tiene que

$$x \hat{\star} y = \texttt{fl}(x \star y), \qquad \star \in \{+, -, \cdot, \div, \sqrt{\ }\}$$

Esto así, se tiene el modelo estándar para los números máquina y sus operaciones:

Para $x, y \in \mathbb{F}$, $\star \in \{+, -, \cdot, \div, \sqrt{\ }\}$ en donde $|\varepsilon| \leq \varepsilon_m$, tal que

4 Operaciones punto flotante

La IEEE 754 define que las operaciones aritméticas y la operación raíz cuadrada son calculadas para números máquina por un redondeo correcto para el resultado exacto que se persigue: si se denota la realización de una operación como $\star$ por $\hat{\star}$, entonces para $x, y \in \mathbb{F}$ se tiene que

$$x \hat{\star} y = \texttt{fl}(x \star y), \qquad \star \in \{+, -, \cdot, \div, \sqrt{\ }\}$$

Esto así, se tiene el modelo estándar para los números máquina y sus operaciones:

Para $x, y \in \mathbb{F}$, $\star \in \{+, -, \cdot, \div, \sqrt{\ }\}$ en donde $|\varepsilon| \leq \varepsilon_m$, tal que

$$x \hat{\star} y = (x \star y)(1 + \varepsilon)$$

4 Operaciones punto flotante

La IEEE 754 define que las operaciones aritméticas y la operación raíz cuadrada son calculadas para números máquina por un redondeo correcto para el resultado exacto que se persigue: si se denota la realización de una operación como $\star$ por $\hat{\star}$, entonces para $x, y \in \mathbb{F}$ se tiene que

$$x \hat{\star} y = \mathtt{fl}(x \star y), \qquad \star \in \{+, -, \cdot, \div, \sqrt{\ }\}$$

Esto así, se tiene el modelo estándar para los números máquina y sus operaciones:

Para $x, y \in \mathbb{F}$, $\star \in \{+, -, \cdot, \div, \sqrt{\cdot}\}$ en donde $|\varepsilon| \leq \varepsilon_m$, tal que

$$x \hat{\star} y = (x \star y)(1 + \varepsilon)$$

para la realización-máquina $\hat{\star}$ de la operación $\star$.

4 Operaciones punto flotante

4 Operaciones punto flotante

La idea es evitar overflow o underflow. Así, las operaciones aritméticas en precisión finita se definen así:

La idea es evitar overflow o underflow. Así, las operaciones aritméticas en precisión finita se definen así:

Definition

La idea es evitar overflow o underflow. Así, las operaciones aritméticas en precisión finita se definen así:

Definition

- **Suma:** $x \oplus y := \mathtt{fl}(\mathtt{fl}(x) + \mathtt{fl}(y))$

4 Operaciones punto flotante

La idea es evitar overflow o underflow. Así, las operaciones aritméticas en precisión finita se definen así:

Definition

- **Suma:** $x \oplus y := \mathtt{fl}(\mathtt{fl}(x) + \mathtt{fl}(y))$
- **Resta:** $x \ominus y := \mathtt{fl}(\mathtt{fl}(x) - \mathtt{fl}(y))$

4 Operaciones punto flotante

La idea es evitar overflow o underflow. Así, las operaciones aritméticas en precisión finita se definen así:

Definition

- **Suma:** $x \oplus y := \mathtt{fl}(\mathtt{fl}(x) + \mathtt{fl}(y))$
- **Resta:** $x \ominus y := \mathtt{fl}(\mathtt{fl}(x) - \mathtt{fl}(y))$
- **Producto:** $x \odot y := \mathtt{fl}(\mathtt{fl}(x) \cdot \mathtt{fl}(y))$

La idea es evitar overflow o underflow. Así, las operaciones aritméticas en precisión finita se definen así:

Definition

- **Suma:** $x \oplus y := \mathtt{fl}(\mathtt{fl}(x) + \mathtt{fl}(y))$
- **Resta:** $x \ominus y := \mathtt{fl}(\mathtt{fl}(x) - \mathtt{fl}(y))$
- **Producto:** $x \odot y := \mathtt{fl}(\mathtt{fl}(x) \cdot \mathtt{fl}(y))$
- **División:** $x \oslash y := \mathtt{fl}(\mathtt{fl}(x) \div \mathtt{fl}(y))$

4 Operaciones punto flotante

La idea es evitar overflow o underflow. Así, las operaciones aritméticas en precisión finita se definen así:

Definition

- **Suma:** $x \oplus y := \texttt{fl}(\texttt{fl}(x) + \texttt{fl}(y))$
- **Resta:** $x \ominus y := \texttt{fl}(\texttt{fl}(x) - \texttt{fl}(y))$
- **Producto:** $x \odot y := \texttt{fl}(\texttt{fl}(x) \cdot \texttt{fl}(y))$
- **División:** $x \oslash y := \texttt{fl}(\texttt{fl}(x) \div \texttt{fl}(y))$

Nota:

La suma y el producto son conmutativas pero no asociativas. Tampoco se cumple la ley distributiva.

¡Muchas Gracias!