Generación de Terrenos con Redes Neuronales Multicapa

Sistemas de Inteligencia Artificial - Instituto Tecnológico de Buenos Aires

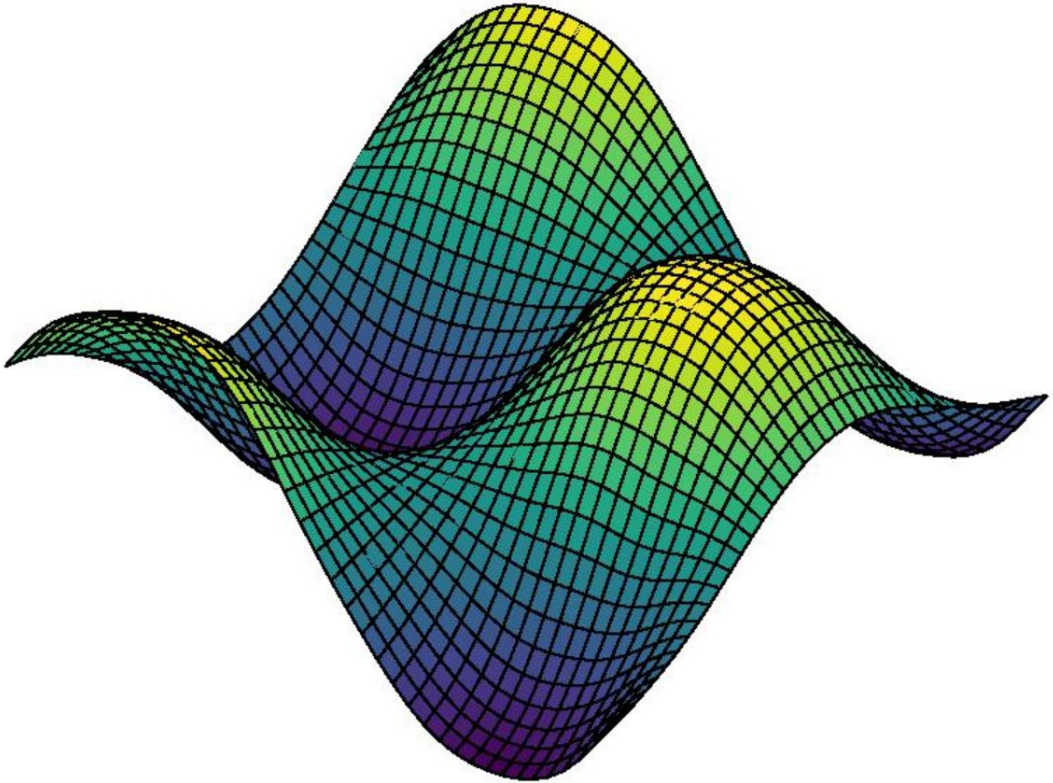
Grupo 8

Garrigó, Mariano
54393

Raies, Tomás
56099

Saqués, Alejo
56047

Terreno



Entrenamiento

Pre-procesamiento de datos:

- Feature Scaling para estandarizar las muestras

$$x_i' = \frac{x_i - min(X)}{max(X) - min(X)}(b - a) + a$$

X: Conjunto de muestras

Entrenamiento

Muestras de entrenamiento tomadas del terreno:

Seleccionadas bajo el siguiente criterio de manera no determinística

- 90% para entrenamiento.
- 10% para testeo.

Implementación: Back-propagation

Algoritmos utilizados:

1. Incremental con *adaptive eta* y *momentum*
2. Batch con *momentum*
3. Batch parametrizable

Parámetros de entrenamiento

- f: Funcion de activacion
- H: Capas ocultas (array)
- η: Learning rate
- *momentum*: Factor del momentum
- α: Constante de crecimiento del eta adaptativo
- β: Factor de decrecimiento del eta adaptativo
- k: Constante de decrecimiento consistente
- ε: Cota inferior para (condición de corte)
- ε_e: Condición de corte del error cuadrático medio
- max_{iters}: Número máximo de epochs.

Análisis de resultados

- Condición de corte (ε_e , *epochs)*
- Tasas de éxito $\text{Épsilon } (\epsilon)\text{:} \quad [S_i - \epsilon, S_i + \epsilon]$
- Diferencia entre valor esperado y emitido $D_i = abs(S_{c_i} - E_i)$
- Error cuadrático medio

Resultados

1. *Alg. Incremental con eta adaptativo*

Resultados - Inc. con eta adaptativo

- $f = tanh$
- $H = [10,10]$
- $\eta = 0,1$
- $momentum = 0$
- $\alpha = 0,05$
- $\beta = 0,1$
- $k = 5$
- $\varepsilon = 0,00001$
- $\varepsilon_e = 0,0001$
- $max_{iters} = 10000$

Resultados - Inc. con eta adaptativo

- Condición de corte:

Se alcanzó el máximo de iteraciones

ECM = $6{,}5 \times 10^{-4}$

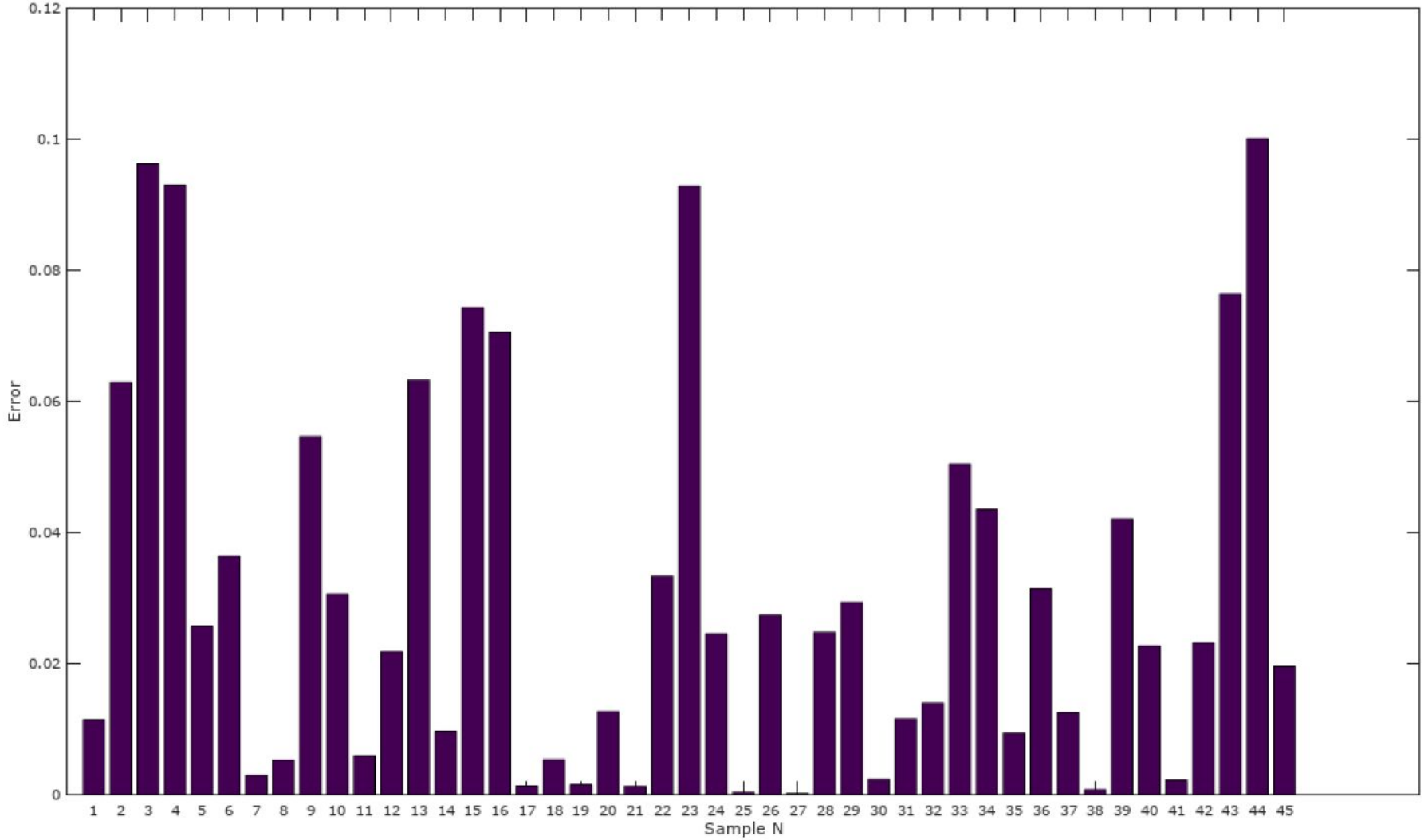
Resultados - Inc. con eta adaptativo

- Tasas de éxito:

Épsilon	Éxito (%)
.05	75.6
.06	80
.07	84.5
.08	91.2
.09	91.2
.1	97.8
.11	100

Resultados - Inc. con eta adaptativo

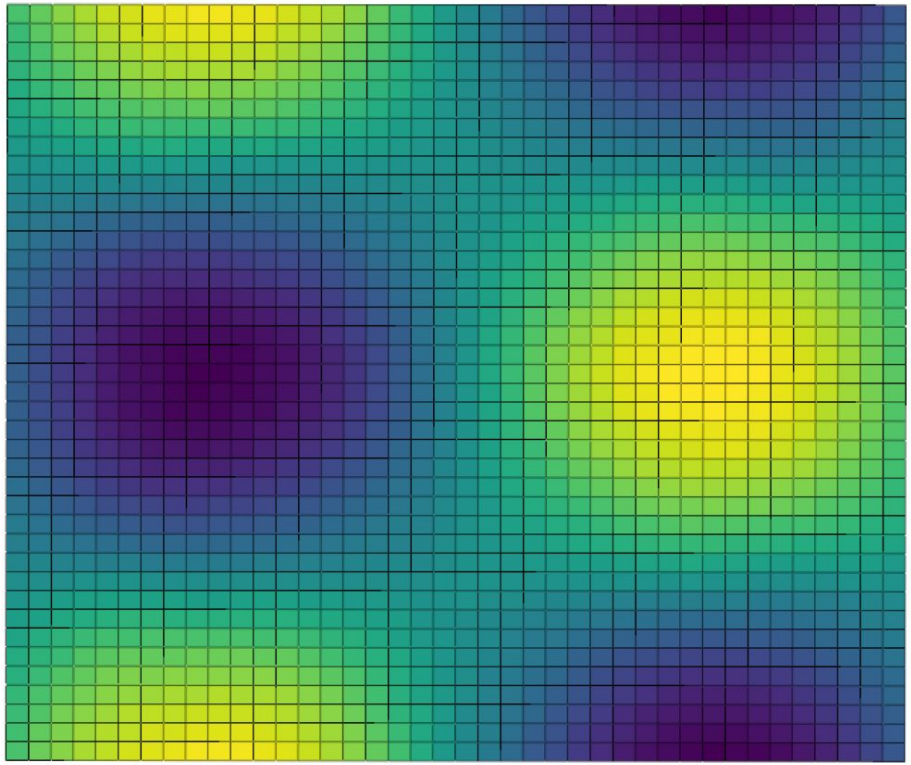
- Diferencia entre valor esperado y emitido:
  - $max(D) = 0{,}10006$
  - $min(D) = 1{,}7035 * 10^{-4}$
  - $mean(D) = 0{,}030684$


0.12
0.1
0.08
0.06
0.04
0.02
0
Error
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45
Sample N

Comportamiento - Picos de error

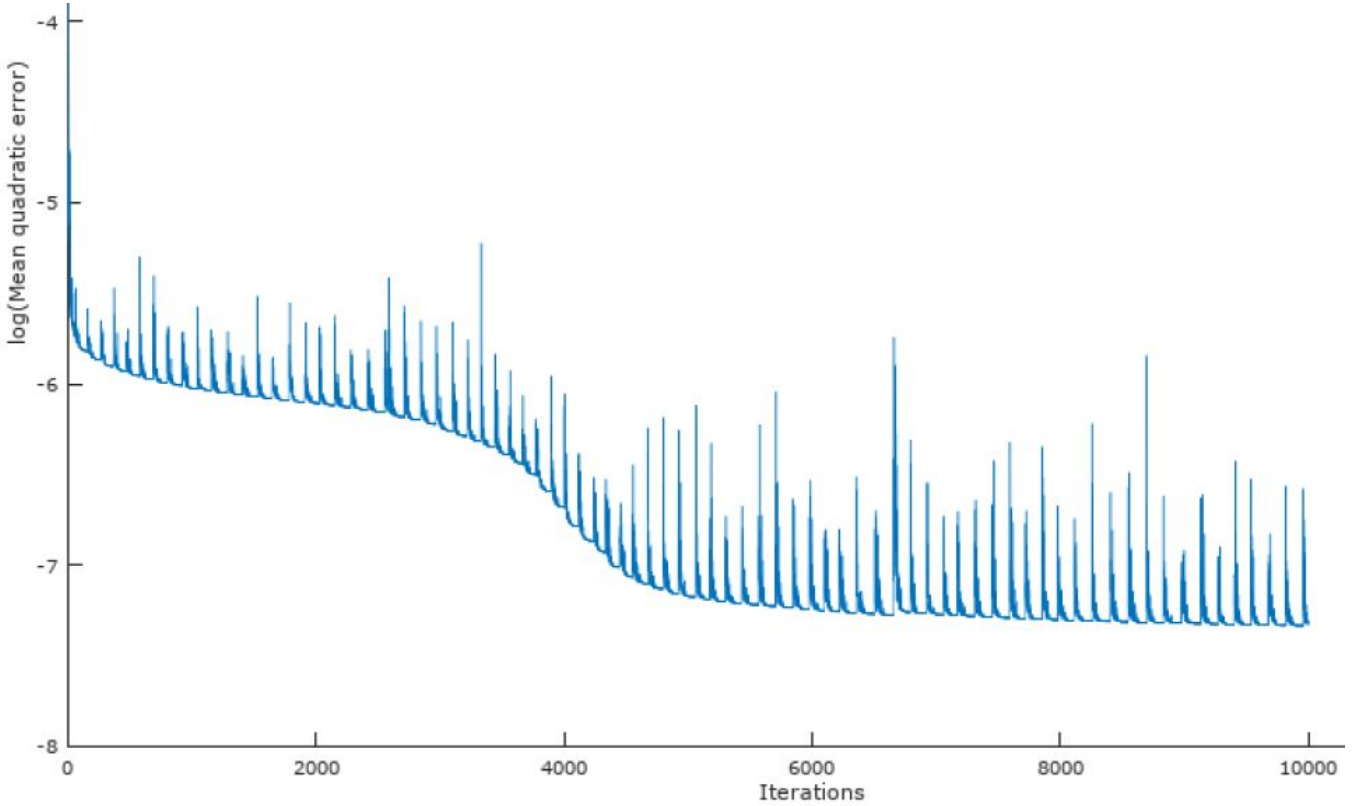
Muestra	X	Y
44	-0.14806	0.31066
3	-1	-0.93007
23	1	-0.11425
4	0.061232	-0.240449

Mapa de Altura



Resultados - Inc. con eta adaptativo

- ECM:



Resultados - Batch con *momentum*

- $f = tanh$
- $H = [10,10]$
- $\eta = 0{,}001$
- $momentum = 0{,}9$
- $\varepsilon_e = 0{,}0001$
- $max_{iters} = 100000$

Resultados - Batch con *momentum*

- Condición de corte:

Se alcanzó el máximo de iteraciones

ECM = $3{,}26 \times 10^{-4}$

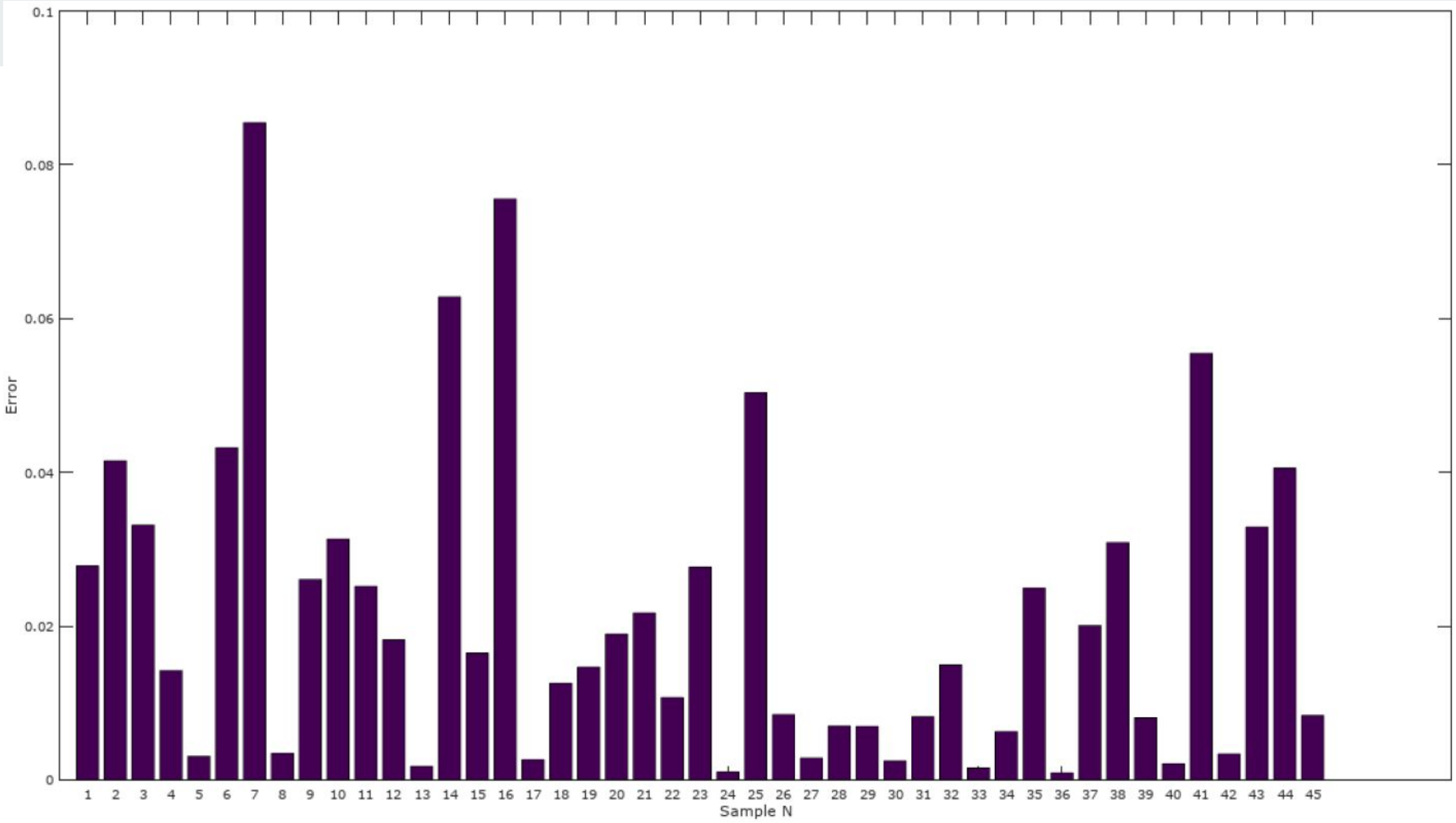
Resultados - Batch con *momentum*

- Tasas de éxito:

Épsilon	Éxito (%)
.05	88.9
.06	93.4
.07	95.6
.08	97.8
.09	100
.1	100

Resultados - Batch con *momentum*

- Diferencia entre valor esperado y emitido:
  - $max(D) = 0{,}085488$
  - $min(D) = 9{,}3254 * 10^{-4}$
  - $mean(D) = 0{,}021261$

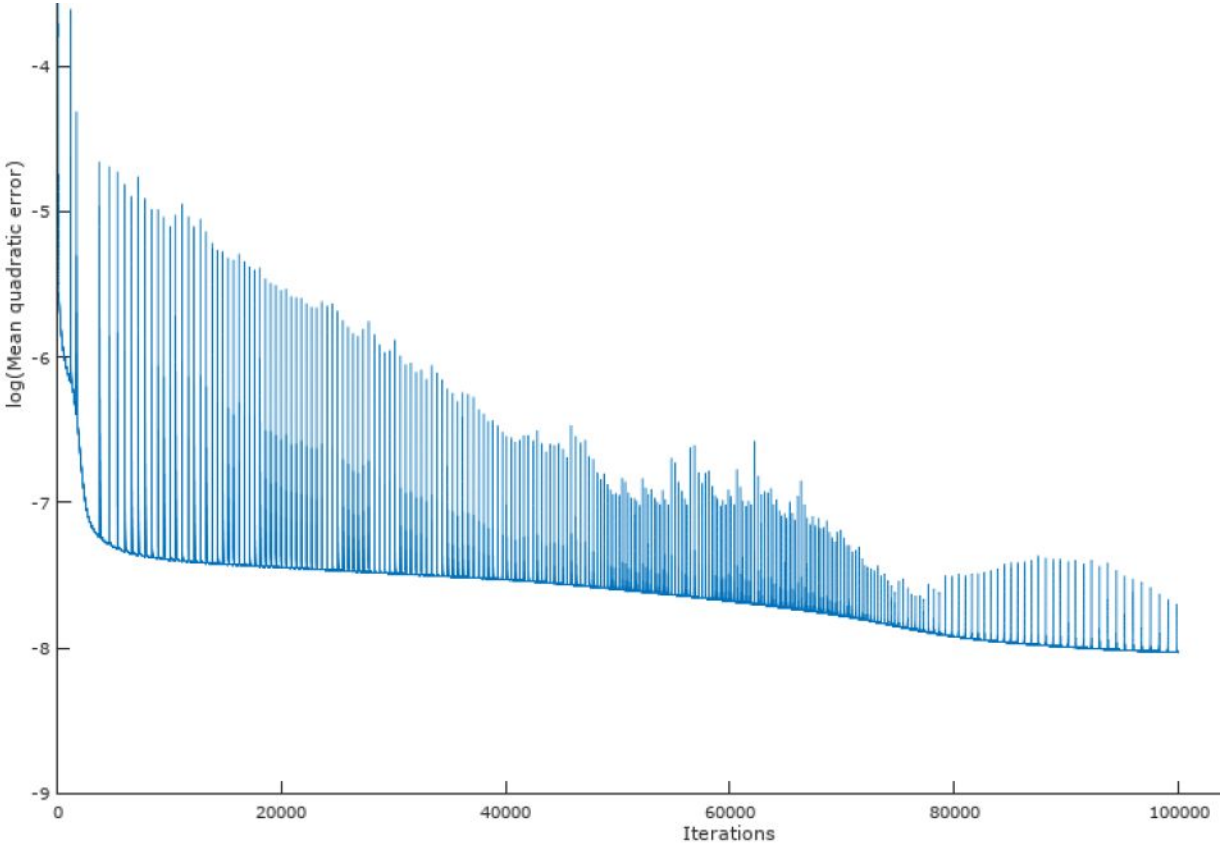

0.1
0.08
0.06
0.04
0.02
0
Error
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45
Sample N

Comportamiento - Picos de error

Muestra	X	Y
7	1	0.58233
16	1	0.82133
14	-0.14806	-0.35267
41	0.94275	-0.24045

Resultados - Batch con *momentum*

- ECM:



Resultados - Batch parametrizable

- $f = tanh$
- $H = [10,10]$
- $\eta = 0{,}001$
- $momentum = 0{,}9$
- $\varepsilon_e = 0{,}0001$
- $max_{iters} = 15000$

Resultados - Batch parametrizable

- Condición de corte:

Se alcanzó el máximo de iteraciones

ECM = $4{,}22 \times 10^{-4}$

Resultados - Batch parametrizable

- Tasas de éxito:

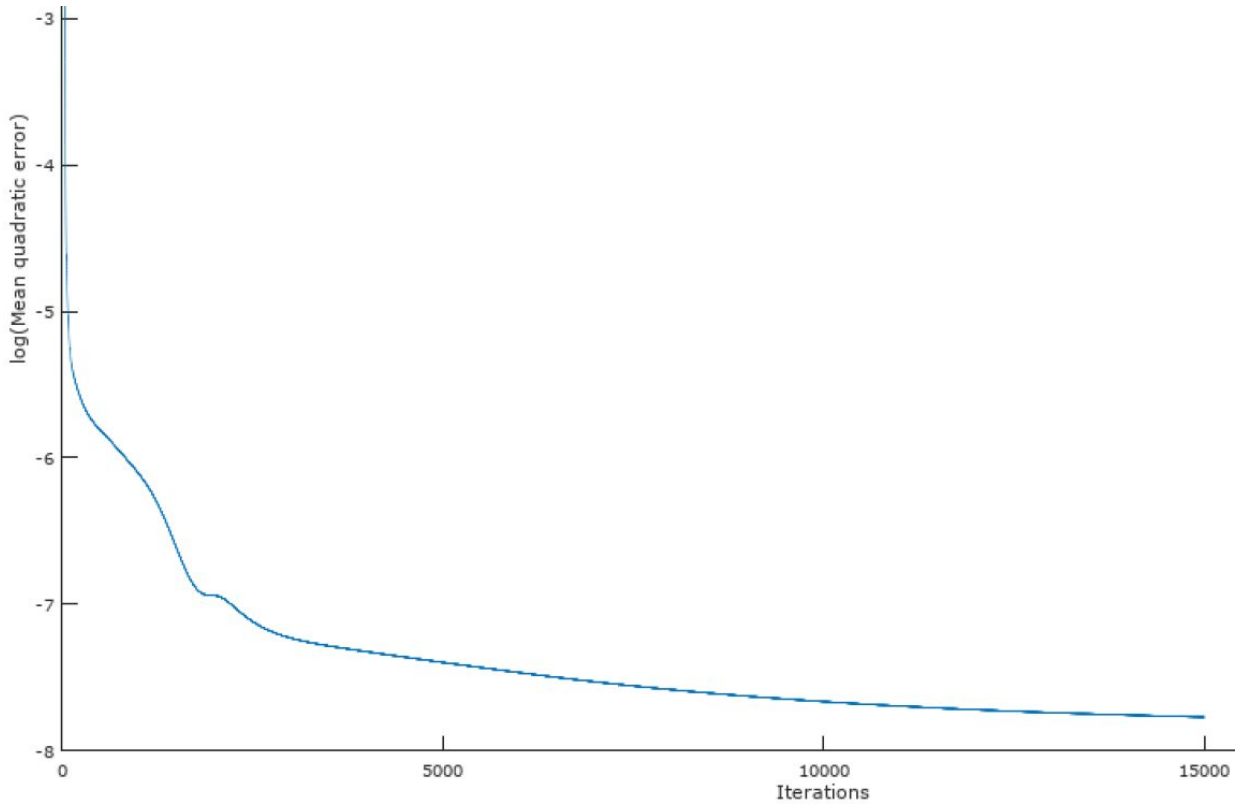
Épsilon	Éxito (%)
.05	77.8
.06	88.9
.07	95.6
.08	97.8
.09	100
.1	100

Resultados - Batch parametrizable

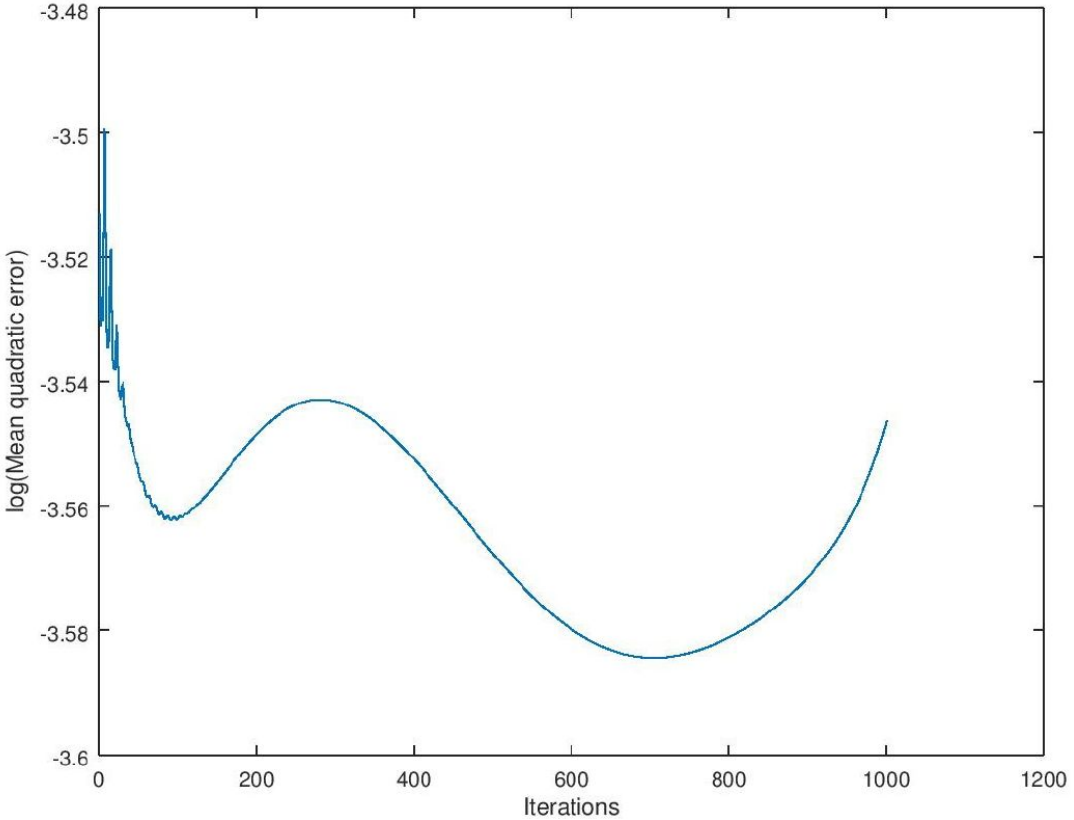
- Diferencia entre valor esperado y emitido:
  - $max(D) = 0{,}084863$
  - $min(D) = 0{,}0011897$
  - $mean(D) = 0{,}028660$

Resultados - Batch parametrizable

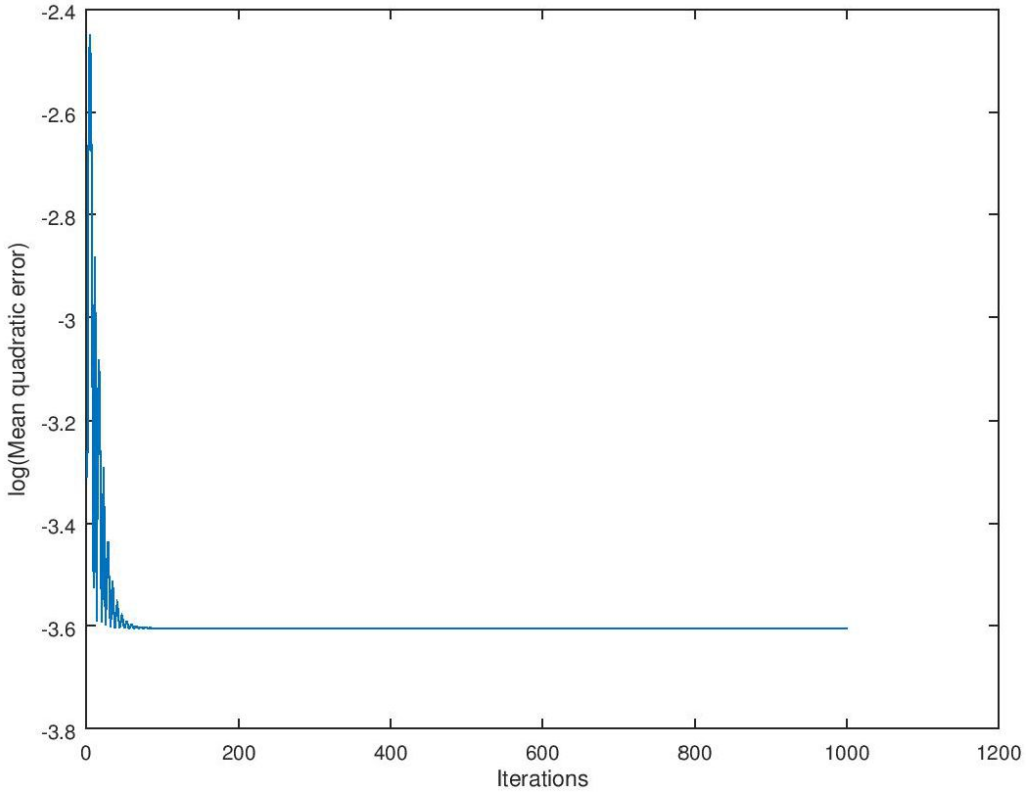
- ECM:



Función logística

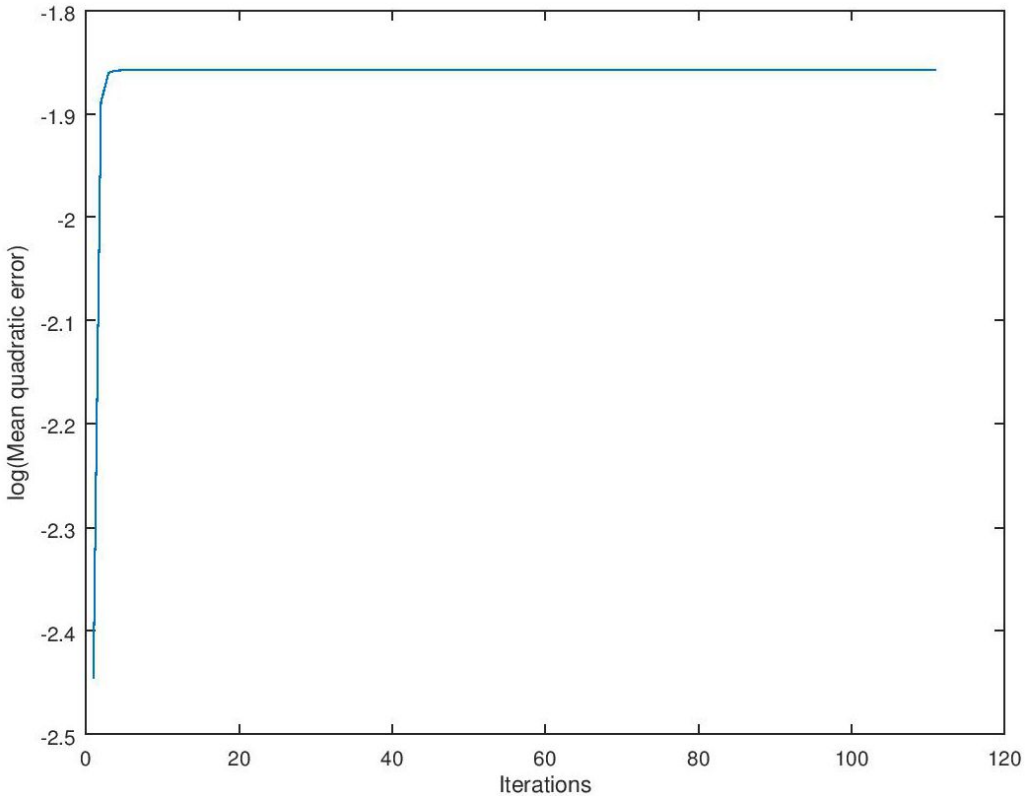


Otras Arquitecturas: Mínimos Locales



- Batch con momentum H[5,5]

Otras Arquitecturas: Incremento del



- Batch con momentum H[80]

Conclusiones

- Las tres redes neuronales propuestas logran resolver el problema.
- En el peor de los casos, se generalizó en torno al 75% de los casos de prueba.
- El algoritmo que ofreció mayor precisión a menor costo computacional fue *Batches Parametrizable*

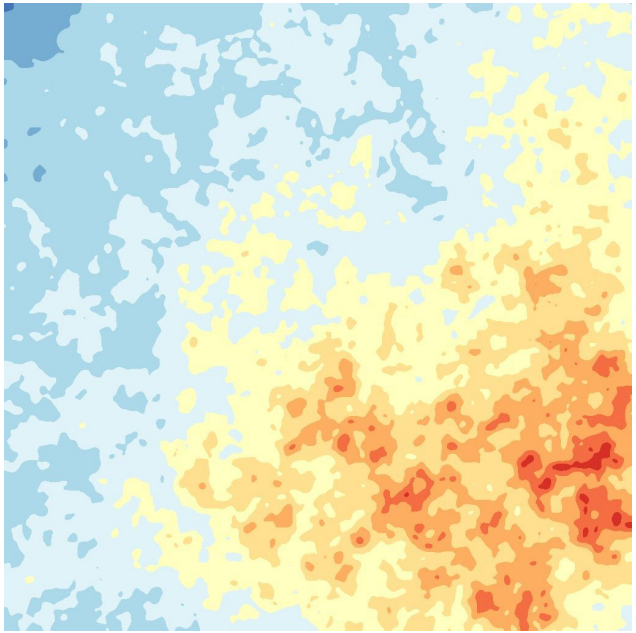
Conclusiones

- *Adaptive eta* fue performante, pero depende mucho de la buena elección de sus parámetros de crecimiento y decrecimiento.
- Notamos que utilizar *momentum* sin *adaptive eta* ofrecen un entrenamiento más estable.

Comparación con otros Algoritmos

- La generación de terrenos procedural es de suma importancia en varias áreas.
- Se utiliza para el desarrollo de videojuegos, permitiendo generar mapas aleatorios bajo ciertos parámetros predefinidos.
- Algunos algoritmos:
  - Diamond Square Algorithm (DSA)
  - Perlin Noise

Ejemplo DSA

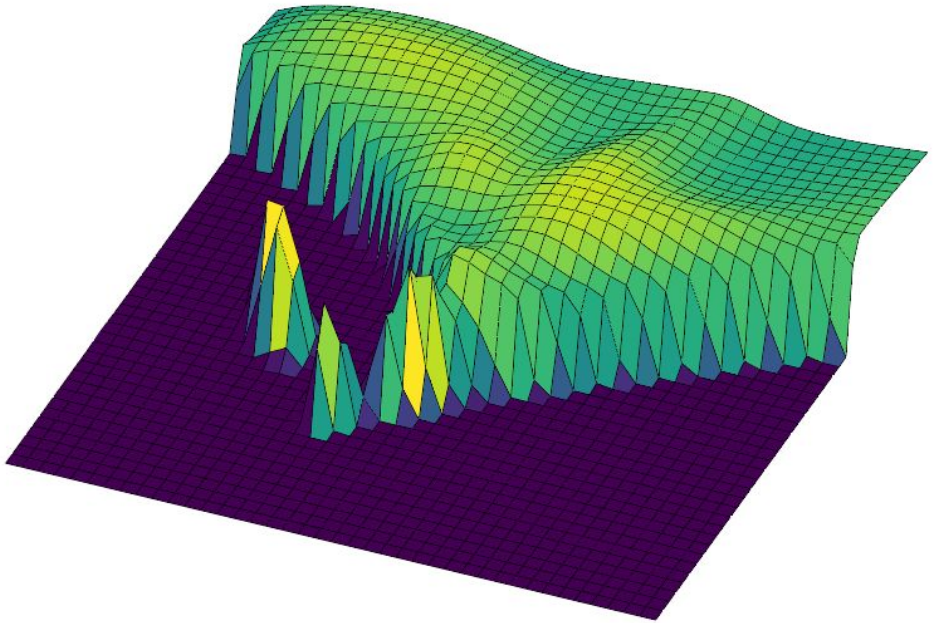


Generación de terreno aleatorio

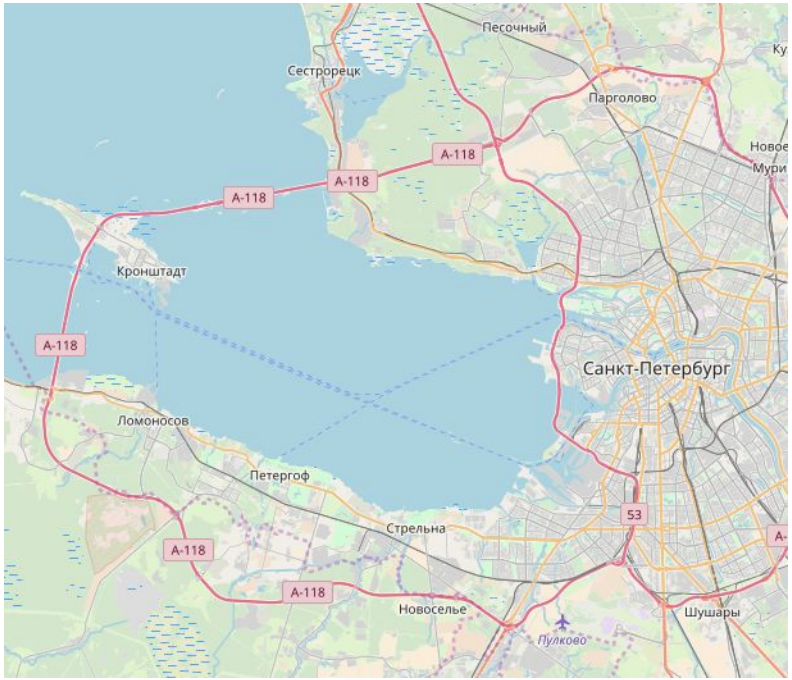
Analizamos el potencial de las redes neuronales multicapa para obtener similares resultados.

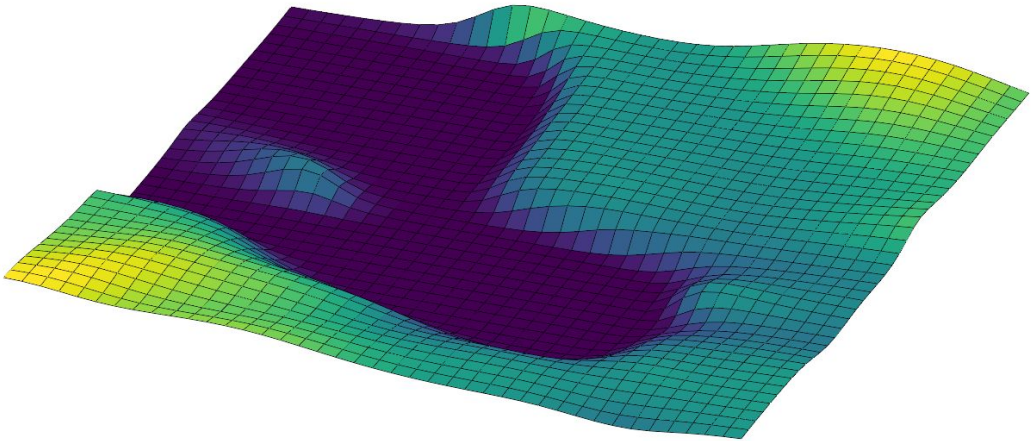
- Tomamos la altura de diversas ciudades del mundo.
- Utilizamos el algoritmo Batch con *momentum*.
- Entrenamos con el 90% de las muestras tomadas.
- Representamos la generalización lograda.

Punta del Este

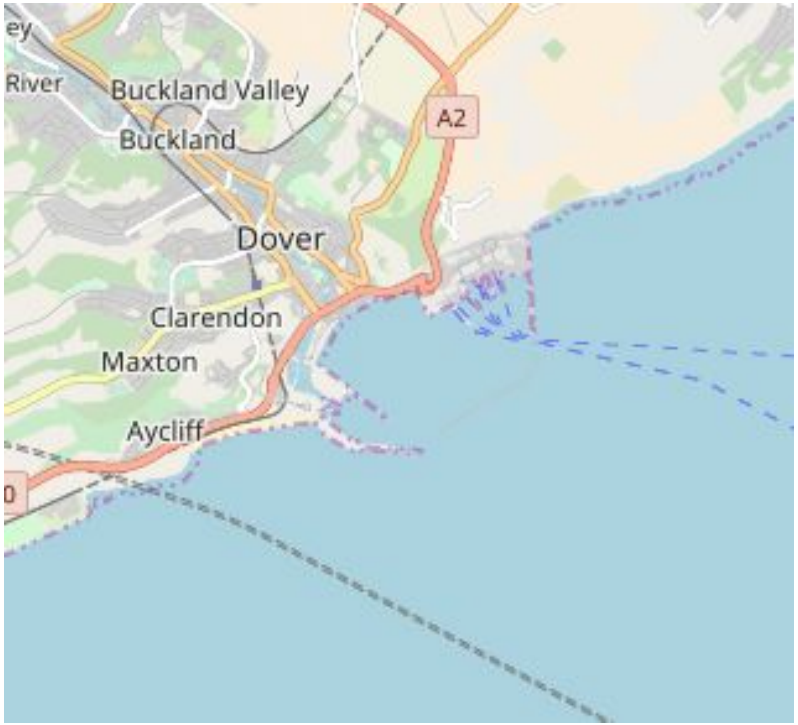


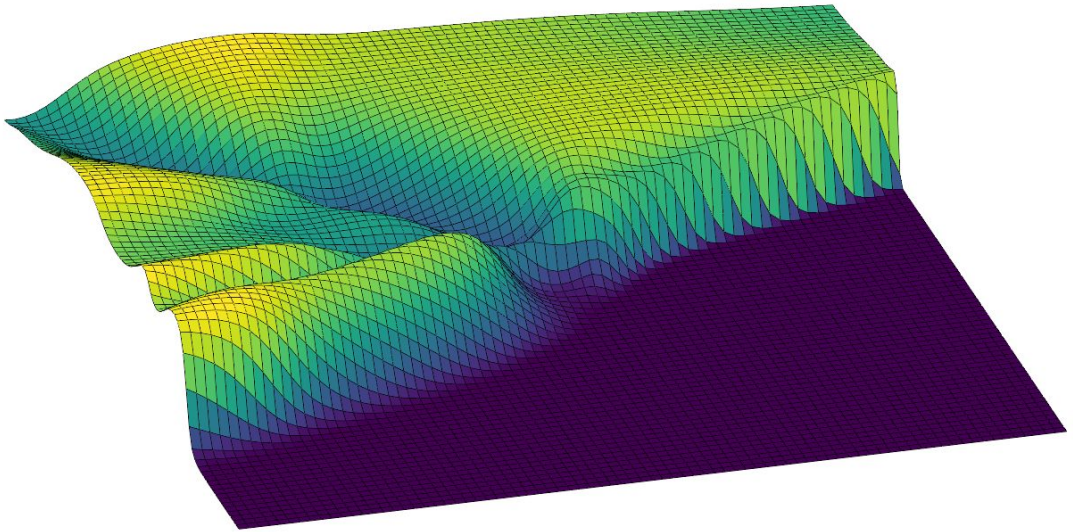
San Petersburgo





Dover


River
Buckland Valley
Buckland
A2
Dover
Clarendon
Maxton
Aycliff



Resultados y conclusiones

- Los órdenes de magnitud de los errores cuadráticos medios son superiores que los obtenidos utilizando el terreno provisto por la cátedra.
- La generalización del terreno real es aceptable, sin embargo DSA tiene mayor capacidad para generar terrenos aleatorios.