Alveolar perfusion and gas transport modelling

Investigación en Pregrado (IPre)

Bastián Herrera

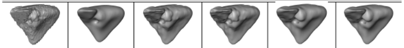
Profesor guía - Daniel Hurtado
Estudiante guía - Pablo Zurita
Pontificia Universidad Católica de Chile

26/05/2022

Tabla de contenidos

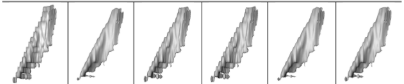
Método lowpass v/s LaplacianHC

Table 2: Smoothing results of the liver surface model by means of different smoothing algorithms with a weighting factor of 0.9 and 50 iterations.



original	Laplace	Laplace+HC	LowPass	Laplace+HC 2nd order	LowPass 2nd order
V=100%	V=91.0% 2.03sec	V=99.9% 3.91sec	V=100.1% 4.36sec	V=99.6% 224.14sec	V=100.2% 220.95sec

Table 3: Smoothing results of the neck muscle model by means of different smoothing algorithms with a weighting factor of 0.5 and 10 iterations.



original	Laplace	Laplace+HC	LowPass	Laplace+HC 2nd order	LowPass 2nd order
V=100%	V=86.5% 0.11sec	V=100.0% 0.23sec	V=99.3% 0.25sec	V=94.4% 12.55sec	V=97.1% 11.97sec

(a) Comparación del suavizado con los métodos lowpass y LaplacianHC, en mallas de hígado y músculo (Bade, Preim & Haase, 2006)

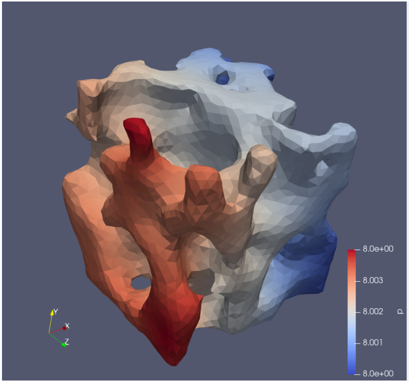
Bade, R., Preim, B. & Haase, J. (2006). Comparison of Fundamental Mesh Smoothing Algorithms for Medical Surface Models. [conferencia] Disponible en `https://www.researchgate.net/publication/221510388_Comparison_of_Fundamental_Mesh_Smoothing_Algorithms_for_Medical_Surface_Models`

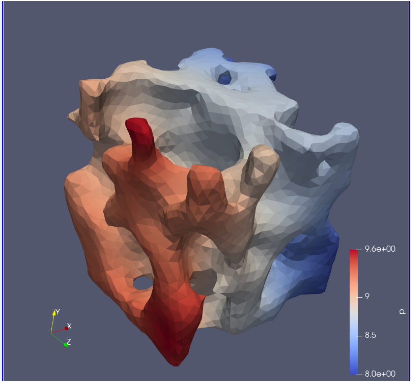
Método lowpass v/s LaplacianHC

```
%% Smoothing
tic

% Sintaxis: sms(nodes, elems, iter, alpha, method)
% alpha = 0.5 luce bien
% method = 'lowpass' es el mejor según documentación, en vez de 'laplacian' y 'laplacianhc'.

sms_fine_5_08 = sms(finenodes, finetrielems, 5, 0.8, 'lowpass');
sms_rough_10_08 = sms(roughnodes, roughtrielems, 10, 0.8, 'lowpass');

toc
```

(b) Método de suavizado utilizado en las mallas de RVEs de prueba en esta presentación.

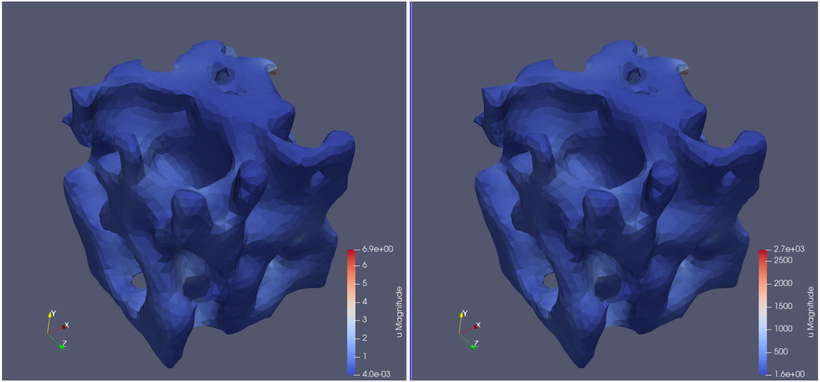
Campo de presiones p





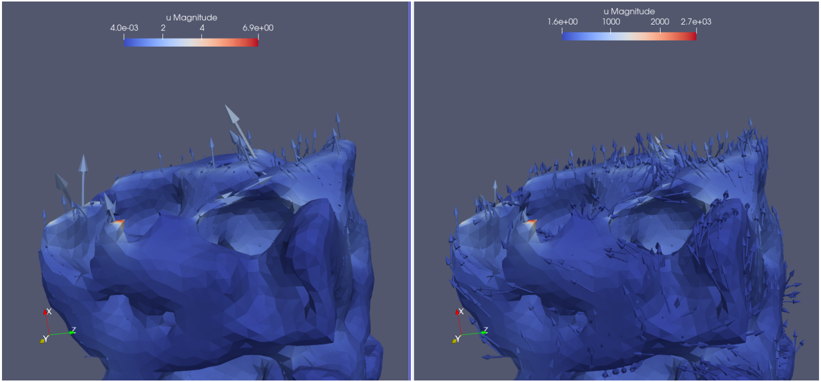
(c) Campo de presiones para los casos $u_{in} = 0.5\,\mu m/s$ (izquierda) y $u_{in} = 200\,\mu m/s$ (derecha).

Campo de velocidades $v = \frac{-1}{\mu}\kappa\nabla p$



(d) Magnitud del campo de velocidades obtenido a partir del campo de presiones p y la ley de Darcy, para los casos $u_{in} = 0.5\,\mu m/s$ (izquierda) y $u_{in} = 200\,\mu m/s$ (derecha).

Campo de velocidades $v = \frac{-1}{\mu}\kappa\nabla p$



(e) Vectores del campo de velocidades, para los casos $u_{in} = 0.5\,\mu m/s$ (izquierda) y $u_{in} = 200\,\mu m/s$ (derecha).

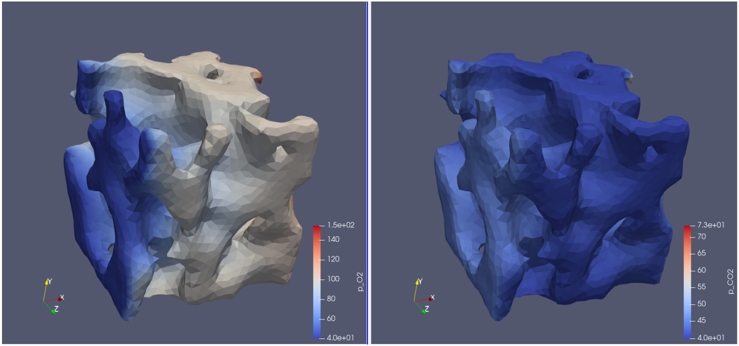
Formulación débil del problema de transporte lineal

The weak, linear formulation for steady blood-side transport that arises in such case is as follows: find $\boldsymbol{s} = (p_{O_2}, p_{CO_2}, c_{HbO_2}, c_{HbCO_2}) \in H^1(\Omega;\mathbb{R}) \times H^1(\Omega;\mathbb{R}) \times L^2(\Omega;\mathbb{R}) \times L^2(\Omega;\mathbb{R})$ such that

$$(WLT)\begin{cases} LG^p_{O_2}(\boldsymbol{s}, v) = 0 & \forall v \in \mathcal{V} \\ LG^p_{CO_2}(\boldsymbol{s}, w) = 0 & \forall w \in \mathcal{V} \\ LG^c_{O_2}(\boldsymbol{s}, \eta) = 0 & \forall \eta \in \mathcal{V} \\ LG^c_{CO_2}(\boldsymbol{s}, \xi) = 0 & \forall \xi \in \mathcal{V} \\ \gamma p_X = p_X^{in} & \text{in } \Gamma_{in}, \text{ for } X \in \{O_2, CO_2\} \\ \gamma c_{HbX} = c_{HbX}^{in} & \text{in } \Gamma_{in}, \text{ for } X \in \{O_2, CO_2\}, \end{cases}$$

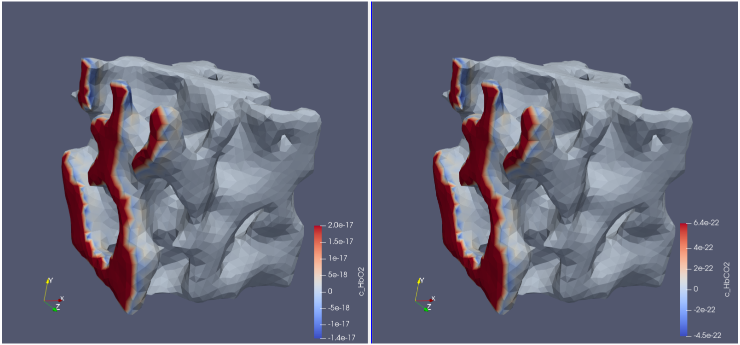
Fig.. Sistema lineal de EDP que modela el intercambio gaseoso (Zurita & Hurtado, 2022)

$u_{in} = 200\ \mu m/s$



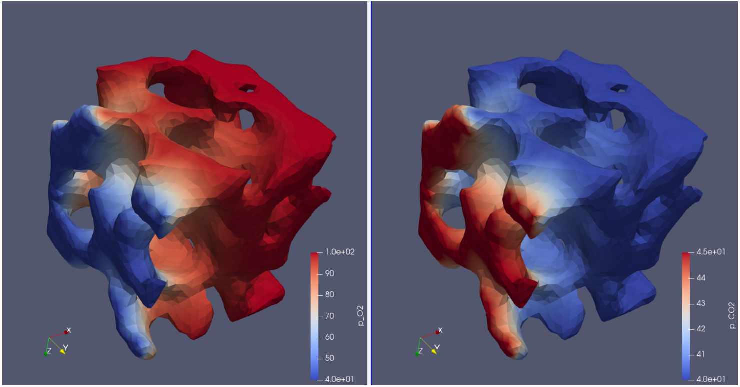
(a) Presiones parciales de O_2 y CO_2 (en mmHg) para el caso $u_{in} = 200\ \mu m/s$, resolviendo el sistema lineal.

$u_{in} = 200\,\mu m/s$



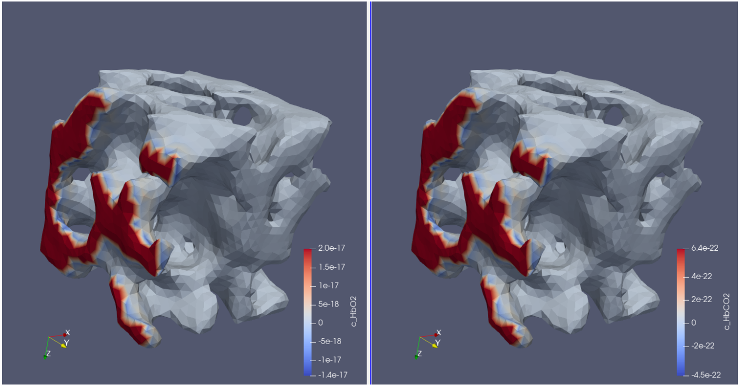
(b) Concentraciones de HbO_2 y $HbCO_2$ (en $mol/\mu m^3$) para el caso $u_{in} = 200\,\mu m/s$, resolviendo el sistema lineal.

$u_{in} = 0.5\,\mu m/s$



(c) Presiones parciales de O_2 y CO_2 (en mmHg) para el caso $u_{in} = 0.5\,\mu m/s$, resolviendo el sistema lineal.

$u_{in} = 0.5\,\mu m/s$



(d) Concentraciones de HbO_2 y $HbCO_2$ (en $mol/\mu m^3$) para el caso $u_{in} = 0.5\,\mu m/s$, resolviendo el sistema lineal.

Formulación débil del problema de transporte no lineal

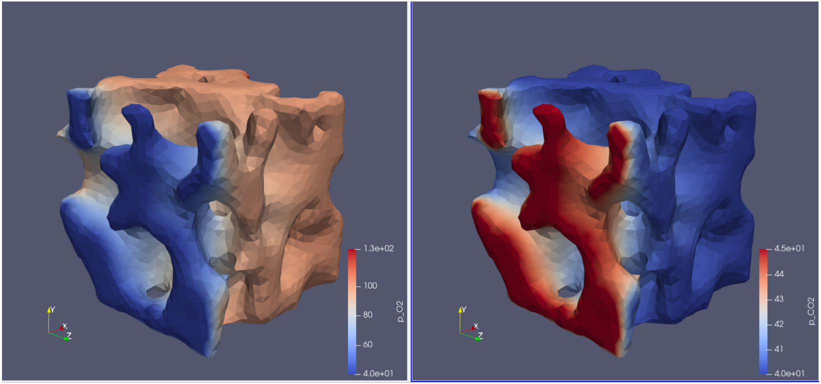
Thus, the weak formulation of the steady blood-side transport problem (WT) reads: find $\boldsymbol{s} = (p_{\mathrm{O}_2}, p_{\mathrm{CO}_2}, c_{\mathrm{HbO}_2}, c_{\mathrm{HbCO}_2}) \in H^1(\Omega;\mathbb{R}) \times H^1(\Omega;\mathbb{R}) \times L^2(\Omega;\mathbb{R}) \times L^2(\Omega;\mathbb{R})$ such that

$$(WT)\begin{cases} G^p_{\mathrm{O}_2}(\boldsymbol{s}, v) = 0 & \forall v \in \mathcal{V} \quad (56a)\\ G^p_{\mathrm{CO}_2}(\boldsymbol{s}, w) = 0 & \forall w \in \mathcal{V} \quad (56b)\\ G^c_{\mathrm{O}_2}(\boldsymbol{s}, \eta) = 0 & \forall \eta \in \mathcal{V} \quad (56c)\\ G^c_{\mathrm{CO}_2}(\boldsymbol{s}, \xi) = 0 & \forall \xi \in \mathcal{V} \quad (56d)\\ \gamma p_{\mathrm{X}} = p^{\mathrm{in}}_{\mathrm{X}} & \text{in } \Gamma_{\mathrm{in}}, \text{ for X} \in \{\mathrm{O}_2, \mathrm{CO}_2\} \quad (56e)\\ \gamma c_{\mathrm{HbX}} = c^{\mathrm{in}}_{\mathrm{HbX}} & \text{in } \Gamma_{\mathrm{in}}, \text{ for X} \in \{\mathrm{O}_2, \mathrm{CO}_2\}. \quad (56f) \end{cases}$$

$$G^c_{\mathrm{X}}(p_{\mathrm{O}_2}, p_{\mathrm{CO}_2}, c_{\mathrm{HbO}_2}, c_{\mathrm{HbCO}_2}, \eta) := -(g_{\mathrm{X}}(p_{\mathrm{X}}, c_{\mathrm{HbX}}, c_{\mathrm{HbY}}), \eta) + (\boldsymbol{u} c_{\mathrm{HbX}}, \nabla\eta) - \langle \boldsymbol{u} c_{\mathrm{HbX}} \cdot \boldsymbol{n}, \eta\rangle_{\Gamma_{\mathrm{out}}} = 0.$$

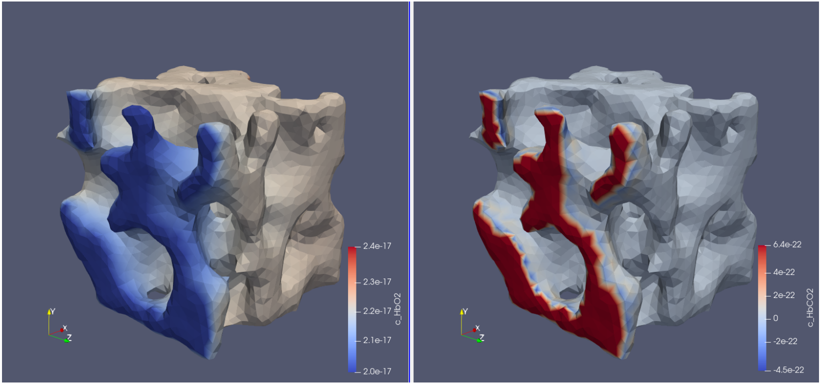
Fig.. Sistema no lineal de EDP que modela el intercambio gaseoso (Zurita & Hurtado, 2022)

$u_{in} = 0.5\,\mu m/s$



(a) Presiones parciales de O_2 y CO_2 (en mmHg) para el caso $u_{in} = 0.5\,\mu m/s$, resolviendo el sistema no lineal.

$u_{in} = 0.5\,\mu m/s$



(b) Concentraciones de HbO_2 y $HbCO_2$ (en $mol/\mu m^3$) para el caso $u_{in} = 0.5\,\mu m/s$, resolviendo el sistema no lineal.

$u_{in} = 200\ \mu m/s$

```
[I 11:22:45.256 LabApp] Saving file at /alveolar-perfusion-transport-modeling/src/model.py
Solving linear variational problem.
No Jacobian form specified for nonlinear variational problem.
Differentiating residual form F to obtain Jacobian J = F'.
Solving nonlinear variational problem.
  Newton iteration 0: r (abs) = 1.168e+07 (tol = 1.000e-08) r (rel) = 1.000e+00 (tol = 1.000e-08)
  Newton iteration 1: r (abs) = 4.119e-08 (tol = 1.000e-08) r (rel) = 3.526e-15 (tol = 1.000e-08)
  Newton solver finished in 1 iterations and 1 linear solver iterations.
No Jacobian form specified for nonlinear variational problem.
Differentiating residual form F to obtain Jacobian J = F'.
Solving nonlinear variational problem.
  Newton iteration 0: r (abs) = 9.989e+08 (tol = 1.000e-08) r (rel) = 1.000e+00 (tol = 1.000e-08)
  Newton iteration 1: r (abs) = 1.743e+08 (tol = 1.000e-08) r (rel) = 1.745e-01 (tol = 1.000e-08)
  Newton iteration 2: r (abs) = 5.533e+11 (tol = 1.000e-08) r (rel) = 5.539e+02 (tol = 1.000e-08)
  Newton iteration 3: r (abs) = 1.417e+11 (tol = 1.000e-08) r (rel) = 1.418e+02 (tol = 1.000e-08)
  Newton iteration 4: r (abs) = 2.173e+12 (tol = 1.000e-08) r (rel) = 2.176e+03 (tol = 1.000e-08)
  Newton iteration 5: r (abs) = 5.437e+11 (tol = 1.000e-08) r (rel) = 5.443e+02 (tol = 1.000e-08)
  Newton iteration 6: r (abs) = 2.488e+12 (tol = 1.000e-08) r (rel) = 2.491e+03 (tol = 1.000e-08)
  Newton iteration 7: r (abs) = 1.298e+12 (tol = 1.000e-08) r (rel) = 1.300e+03 (tol = 1.000e-08)
  Newton iteration 8: r (abs) = 4.487e+11 (tol = 1.000e-08) r (rel) = 4.492e+02 (tol = 1.000e-08)
  Newton iteration 9: r (abs) = 7.932e+11 (tol = 1.000e-08) r (rel) = 7.941e+02 (tol = 1.000e-08)
  Newton iteration 10: r (abs) = 4.878e+13 (tol = 1.000e-08) r (rel) = 4.884e+04 (tol = 1.000e-08)
  Newton iteration 11: r (abs) = 1.996e+13 (tol = 1.000e-08) r (rel) = 1.999e+04 (tol = 1.000e-08)
  Newton iteration 12: r (abs) = 5.112e+12 (tol = 1.000e-08) r (rel) = 5.118e+03 (tol = 1.000e-08)
  Newton iteration 13: r (abs) = 1.273e+12 (tol = 1.000e-08) r (rel) = 1.275e+03 (tol = 1.000e-08)
[I 11:24:00.231 LabApp] Saving file at /alveolar-perfusion-transport-modeling/tests/RVE.ipynb
  Newton iteration 14: r (abs) = 4.033e+11 (tol = 1.000e-08) r (rel) = 4.038e+02 (tol = 1.000e-08)
  Newton iteration 15: r (abs) = 2.848e+12 (tol = 1.000e-08) r (rel) = 2.851e+03 (tol = 1.000e-08)
  Newton iteration 16: r (abs) = 3.529e+12 (tol = 1.000e-08) r (rel) = 3.533e+03 (tol = 1.000e-08)
  Newton iteration 17: r (abs) = 2.618e+14 (tol = 1.000e-08) r (rel) = 2.621e+05 (tol = 1.000e-08)
  Newton iteration 18: r (abs) = 6.539e+13 (tol = 1.000e-08) r (rel) = 6.547e+04 (tol = 1.000e-08)
  Newton iteration 19: r (abs) = 1.630e+13 (tol = 1.000e-08) r (rel) = 1.631e+04 (tol = 1.000e-08)
  Newton iteration 20: r (abs) = 4.066e+12 (tol = 1.000e-08) r (rel) = 4.071e+03 (tol = 1.000e-08)
  Newton iteration 21: r (abs) = 1.198e+12 (tol = 1.000e-08) r (rel) = 1.199e+03 (tol = 1.000e-08)
  Newton iteration 22: r (abs) = 5.675e+11 (tol = 1.000e-08) r (rel) = 5.681e+02 (tol = 1.000e-08)
  Newton iteration 23: r (abs) = 9.103e+12 (tol = 1.000e-08) r (rel) = 9.113e+03 (tol = 1.000e-08)
  Newton iteration 24: r (abs) = 2.342e+12 (tol = 1.000e-08) r (rel) = 2.345e+03 (tol = 1.000e-08)
  Newton iteration 25: r (abs) = 2.954e+12 (tol = 1.000e-08) r (rel) = 2.958e+03 (tol = 1.000e-08)
```

(c) Primeras 25 iteraciones del método de Newton para el caso $u_{in} = 200\ \mu m/s$.

$u_{in} = 200\,\mu m/s$

```
Newton iteration 25: r (abs) = 2.954e+12 (tol = 1.000e-08) r (rel) = 2.958e+03 (tol = 1.000e-08)
Newton iteration 26: r (abs) = 1.994e+14 (tol = 1.000e-08) r (rel) = 1.996e+05 (tol = 1.000e-08)
Newton iteration 27: r (abs) = 5.074e+13 (tol = 1.000e-08) r (rel) = 5.080e+04 (tol = 1.000e-08)
Newton iteration 28: r (abs) = 1.266e+13 (tol = 1.000e-08) r (rel) = 1.267e+04 (tol = 1.000e-08)
Newton iteration 29: r (abs) = 3.154e+12 (tol = 1.000e-08) r (rel) = 3.158e+03 (tol = 1.000e-08)
Newton iteration 30: r (abs) = 2.875e+12 (tol = 1.000e-08) r (rel) = 2.878e+03 (tol = 1.000e-08)
Newton iteration 31: r (abs) = 2.833e+12 (tol = 1.000e-08) r (rel) = 2.836e+03 (tol = 1.000e-08)
Newton iteration 32: r (abs) = 4.809e+12 (tol = 1.000e-08) r (rel) = 4.814e+03 (tol = 1.000e-08)
Newton iteration 33: r (abs) = 1.289e+12 (tol = 1.000e-08) r (rel) = 1.291e+03 (tol = 1.000e-08)
Newton iteration 34: r (abs) = 1.456e+12 (tol = 1.000e-08) r (rel) = 1.458e+03 (tol = 1.000e-08)
Newton iteration 35: r (abs) = 1.272e+13 (tol = 1.000e-08) r (rel) = 1.274e+04 (tol = 1.000e-08)
Newton iteration 36: r (abs) = 8.151e+12 (tol = 1.000e-08) r (rel) = 8.161e+03 (tol = 1.000e-08)
Newton iteration 37: r (abs) = 2.028e+12 (tol = 1.000e-08) r (rel) = 2.030e+03 (tol = 1.000e-08)
Newton iteration 38: r (abs) = 5.589e+11 (tol = 1.000e-08) r (rel) = 5.595e+02 (tol = 1.000e-08)
Newton iteration 39: r (abs) = 5.600e+11 (tol = 1.000e-08) r (rel) = 5.607e+02 (tol = 1.000e-08)
Newton iteration 40: r (abs) = 1.058e+12 (tol = 1.000e-08) r (rel) = 1.060e+03 (tol = 1.000e-08)
Newton iteration 41: r (abs) = 2.473e+14 (tol = 1.000e-08) r (rel) = 2.476e+05 (tol = 1.000e-08)
Newton iteration 42: r (abs) = 6.180e+13 (tol = 1.000e-08) r (rel) = 6.187e+04 (tol = 1.000e-08)
Newton iteration 43: r (abs) = 1.537e+13 (tol = 1.000e-08) r (rel) = 1.539e+04 (tol = 1.000e-08)
Newton iteration 44: r (abs) = 3.778e+12 (tol = 1.000e-08) r (rel) = 3.783e+03 (tol = 1.000e-08)
Newton iteration 45: r (abs) = 1.054e+12 (tol = 1.000e-08) r (rel) = 1.055e+03 (tol = 1.000e-08)
Newton iteration 46: r (abs) = 9.228e+11 (tol = 1.000e-08) r (rel) = 9.239e+02 (tol = 1.000e-08)
Newton iteration 47: r (abs) = 9.418e+11 (tol = 1.000e-08) r (rel) = 9.429e+02 (tol = 1.000e-08)
Newton iteration 48: r (abs) = 3.410e+11 (tol = 1.000e-08) r (rel) = 3.414e+02 (tol = 1.000e-08)
Newton iteration 49: r (abs) = 1.642e+13 (tol = 1.000e-08) r (rel) = 1.643e+04 (tol = 1.000e-08)
Newton iteration 50: r (abs) = 4.065e+12 (tol = 1.000e-08) r (rel) = 4.069e+03 (tol = 1.000e-08)
```

(d) Últimas 25 iteraciones del método de Newton para el caso $u_{in} = 200\,\mu m/s$. Se observa la no convergencia del método en este caso.

Razones posibles de no convergencia

1. Tolerancia de 10^{-8}, quizás no ajustada al tamaño de malla o a magnitud de u_{in}
2. Parámetro de relajación α en las iteraciones de Newton

The discretized Newton scheme corresponds to constructing a sequence of functions $\left\{s_h^n\right\}_n$ defined according to

$$s_h^n := s_h^{n-1} + \delta s_h^{n-1}, \tag{87}$$

where δs_h^{n-1} is the function associated to a vector solution $\delta \mathbf{s}_h$ of the discretised problem described earlier, given $s = s_h^{n-1}$. Such an iteration has, at the moment, no guarantees for convergence. As a heuristic, we define the initial approximation as

$$s_h^0 := s_h^L, \tag{88}$$

$$\rightarrow s_h^n = s_h^{n-1} + \alpha^{n-1} \delta s_h^{n-1}$$

3. Guess inicial para el problema con `hb=True` no suficientemente cerca de la solución