

# A Mixed Finite Volume Method



*Université de Pau  
et des  
Pays de l'Adour*

David Trujillo

Université de Pau et des Pays de l'Adour  
Laboratoire de Mathématiques Appliquées

# Outline of the talk

- **The mixed finite volume method**
  - History
  - The elliptic case (the flow equation)
  - The convection–diffusion case (the nuclide transport)
- **Implementation in Diffpack**
- **Numerical result**
  - The Pressure
  - Darcy Velocities
  - Nuclide Transport

# The mixed finite volume method

## ■ History

- (1994) Phd Thesis directed by J.M. Thomas  
Oil recovery simulator : 3D, Black Oil, Fortran 77
- (1998) : Drying Process Simulator (C++)  
With the *Laboratoire génie des procédés de Pau*  
ICIAM'99
- (1999) J.M. Thomas , D. T. : *Mixed Finite Volume Method*  
Int. Journal of Num. Meth. in Eng.
- (2000–2001) : Fluvial Hydrodynamic (Diffpack)  
The class FiniteVolume is implemented in Diffpack

## ■ The elliptic case

We consider the problem :

$$\begin{aligned} & -\operatorname{div}(K \operatorname{grad} p) = f, \quad \text{on } \Omega. \\ & + \text{Boundary conditions} \end{aligned}$$

We introduce the vectorial unknown:  $u = -K \operatorname{grad} p$

The problem becomes

$$\begin{cases} u = -K \operatorname{grad} p \text{ on } \Omega \\ \operatorname{div} u = f \text{ on } \Omega \end{cases}$$

**The mixed primal dual formulation of this problem is:**

$u \in H(\operatorname{div}, \Omega), p \in H_0^1(\Omega)$  such that

$$a(u, v) \longrightarrow \begin{cases} \int_{\Omega} u \cdot v \, dx + \int_{\Omega} K \operatorname{grad} p \cdot v \, dx = 0, \forall v \in (L^2(\Omega))^2 \\ \int_{\Omega} q \operatorname{div} u \, dx = \int_{\Omega} f q \, dx, \forall q \in L^2(\Omega) \end{cases}$$

$b_1(p, v)$

$b_2(u, q)$

**Recall that the classical dual formulation is given by :**

$u \in H(\operatorname{div}, \Omega), p \in L^2(\Omega)$  such that

$$\begin{cases} \int_{\Omega} K^{-1} u \cdot v \, dx - \int_{\Omega} p \operatorname{div} v \, dx = 0, \forall v \in H(\operatorname{div}, \Omega) \\ \int_{\Omega} q \operatorname{div} u \, dx = \int_{\Omega} f q \, dx, \forall q \in L^2(\Omega) \end{cases}$$

**In order to use the abstract theory,  
we introduce a general framework**

**The problem becomes**

$$u \in W_1 = H(\operatorname{div}, \Omega), p \in M_1 = H^1(\Omega)$$

$$\begin{cases} a(u, v) + b_1(q, v) = l_1(q), \forall v \in W_2 = (L^2(\Omega))^2 \\ b_2(u, q) = l_2(q), \forall q \in M_2 = L^2(\Omega) \end{cases}$$

**We have 3 inf-sup conditions to verify.**

- $\inf_{u \in W_1} \sup_{q \in M_2} b_2(u, q) > \beta_2 \|u\|_{W_1} \|q\|_{M_2}$

- $\inf_{p \in M_2} \sup_{v \in V_2} b_1(p, v) > \beta_1 \|p\|_{M_2} \|v\|_{V_2}$

- $\inf_{u \in V_1} \sup_{v \in W_2} a(u, v) > \alpha \|u\|_{V_1} \|v\|_{W_2}$

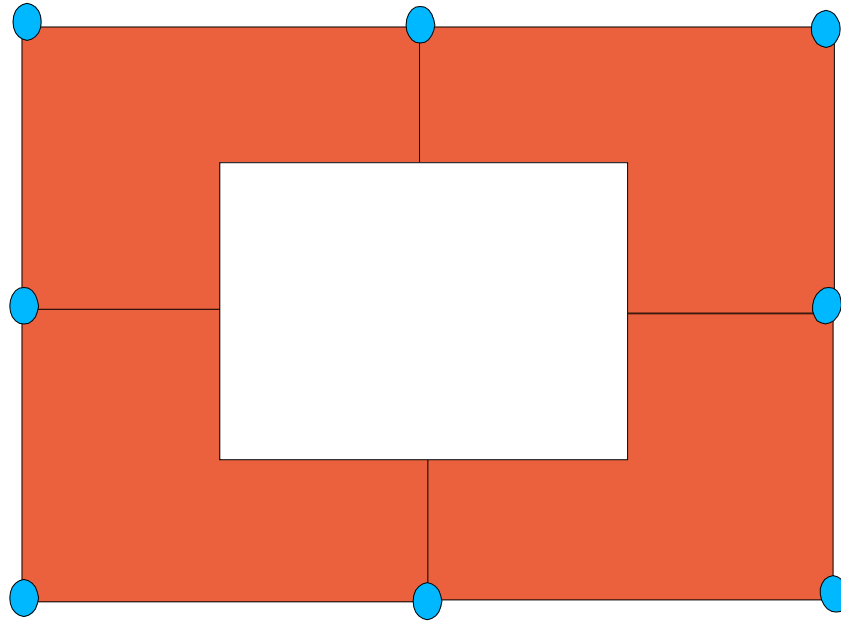
**with**  $V_1 = \{u \in W_1, b_2(u, q) = 0, \forall q \in M_2\}$

$V_2 = \{v \in W_2, a(u, v) = 0, \forall u \in V_1\}$

**Nicolaides (1982)**

- **Discretization spaces**

## Unknown functions

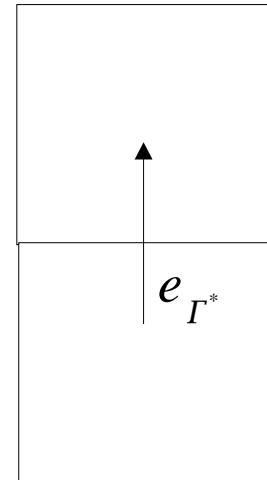
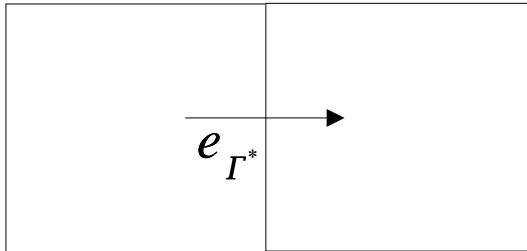


$$M_{1h} = \{p_h \in C^0(\bar{\Omega}); \forall K \in T_h, p_h|_K \in Q_1(K)\}$$

$$W_{1h} = \{u_h \in H(\text{div}, \Omega); \forall K^* \in T_h^*, u_h \in RT_0(K^*)\}$$

## Test functions

$$M_{2h} = \{q_h \in L^2(\Omega); \forall K^* \in T_h^*, q_h|_{K^*} \in P_0(K^*)\}$$



$$W_{2h} = \text{Vect} \{e_{\Gamma^*}, \Gamma^* \in \partial T_h^*\}$$

$\partial T_h^*$  = set of edges of elements  $K^*$  of  $T_h^*$

- **A priori error estimates**

**With this choice of spaces,**

**the discrete problem has a unique solution**

**and,**

$$\|u - u_h\|_{H(\text{div}, \Omega)} + \|p - p_h\|_{1, \Omega} \leq ch.$$

**In the case  $f \in H^1(\Omega)$ , we can prove that**

$$\|u - u_h\|_{0, \Omega} + \|p - p_h\|_{0, \Omega} \leq ch^2$$

- **A posteriori error estimates**

**In the case where  $f$  is constant on each volume**

**we have:**

$$\| \| u - u_h \| \| ^2 + \llbracket p - p_h \rrbracket ^2 = \| K^{-1/2} u_h - K^{1/2} \nabla p_h \|_{0, \Omega}^2,$$

**where**

$$\| \| u - u_h \| \| ^2 = \int_{\Omega} (K^{-1/2} (u - u_h))^2 dx,$$

**and**

$$\llbracket p - p_h \rrbracket ^2 = \int_{\Omega} (K^{1/2} \nabla (p - p_h))^2 dx$$

## ■ The convection–diffusion case

Since  $\operatorname{div} u = 0$ , we consider the next term

$$\operatorname{div}(cu) = \nabla c \cdot u.$$

Here we introduce the vectorial function  $w$  define by:

$$w = -D \nabla c + cu$$

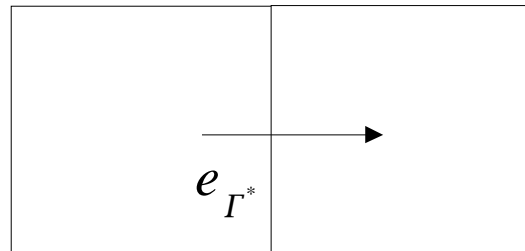
$D$  : effective diffusion/dispersion tensor

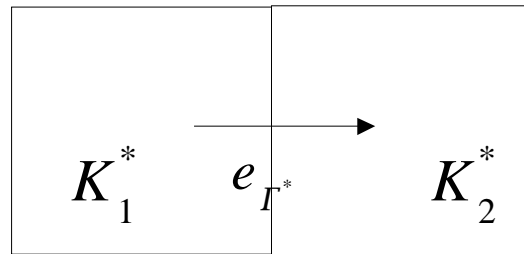
**The mixed formulation becomes**

$$\begin{cases} \int_{\Omega} w \cdot v \, dx + \int_{\Omega} (D \nabla c - cu) \cdot v \, dx = 0, \forall v \in (L^2(\Omega))^2 \\ \int_{\Omega} q \operatorname{div} u \, dx = 0, \forall q \in L^2(\Omega) \end{cases}$$

**Let us now detail the discretization of term :**

$$\int_{\Omega} c u \cdot v \, dx$$





$$\int_{\Gamma^*} c_h u_h \cdot e_{\Gamma^*} dx = \text{meas}(K_1^* \cup K_2^*) c_{\text{amount}} (u_h \cdot e_{\Gamma^*}) / \Gamma^*$$

**Where**

$$c_{\text{amount}} = \begin{cases} c_{K_1^*} & \text{if } (u_h \cdot e_{\Gamma^*}) / \Gamma^* > 0 \\ c_{K_2^*} & \text{otherwise} \end{cases}$$

# Implementation in Diffpack

**Diffpack is a numerical library consisting of C++ classes**

**Class FEM : General description of the FE method.**

**Class FiniteElement : Description of a FE.**

**Contents the element type.**

**More than 35 different finite element types are defined.**

*Remark :* **The description of a FE also contents the integration rule on the element**

## Element–By–Element formulation

$$\int_{\Omega} \nabla p \nabla q \, dx = \sum_{T \in T_h} \int_T \nabla p \cdot \nabla q \, dx = \sum_{T \in T_h} \sum_{a_i} \sum_{j=1}^{nsd} w_i \frac{\partial p}{\partial x_j}(a_i) \frac{\partial q}{\partial x_j}(a_i).$$

The assembly process and the numerical integration are administered by a function `makeSystem` in class `FEM`.

The heart of the `Diffpack` program is coded like this :

**nsd: number of space dimension**

**nbf : number of basis function on the element**

```
for (l=1;l<=nbf;l++)
```

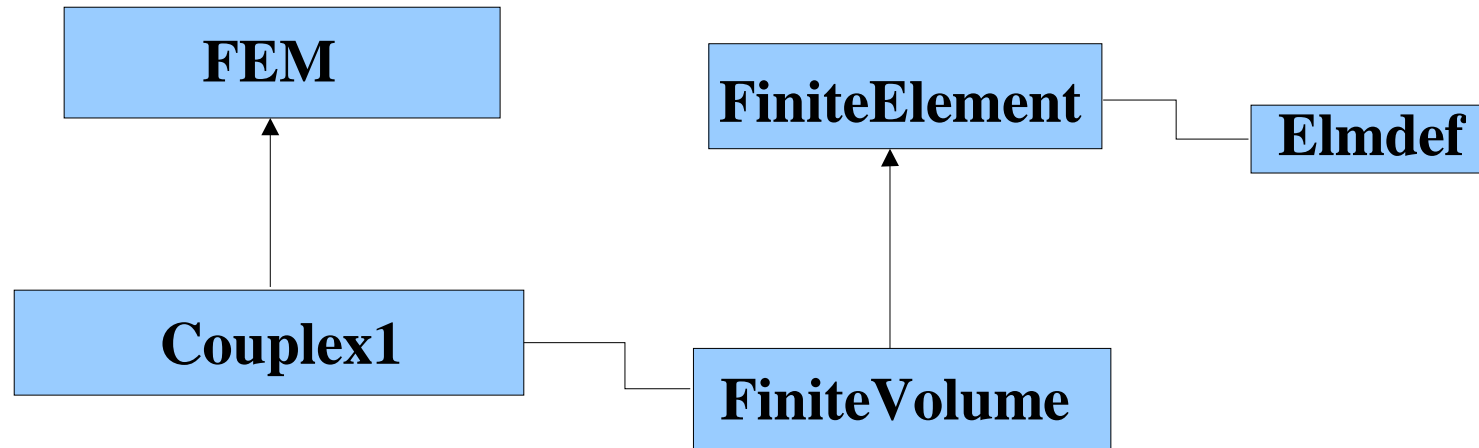
```
  for (m=1; m<=nbf ; m++)
```

```
    for (s=1; s<=nsd; s++)
```

```
      elmat.A(l,m) += fe.dN(l,s)*fe.dN(m,s)
```

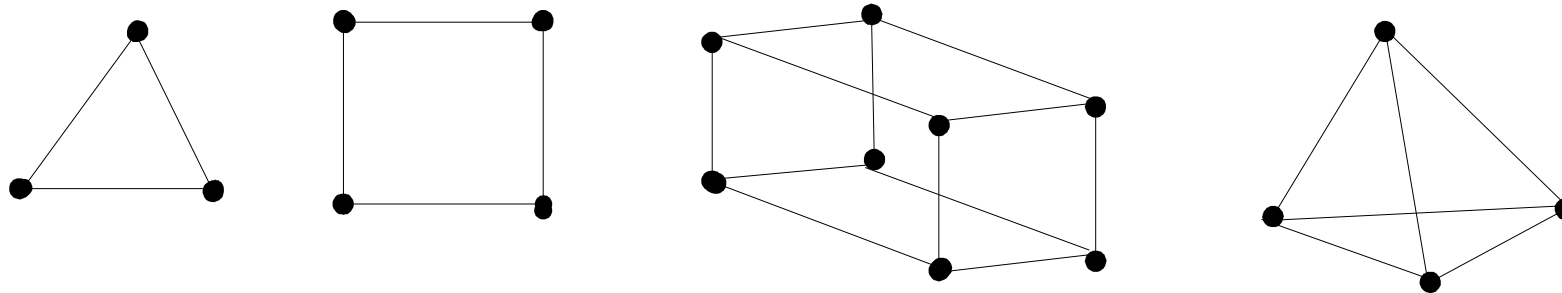
**fe.dN(l,s) =  $\frac{\partial N_l}{\partial x_s}$  at the current integration point**

## The simulator Class must be a subclass of FEM



**Elmdef** : contents the element type which is defined dynamically

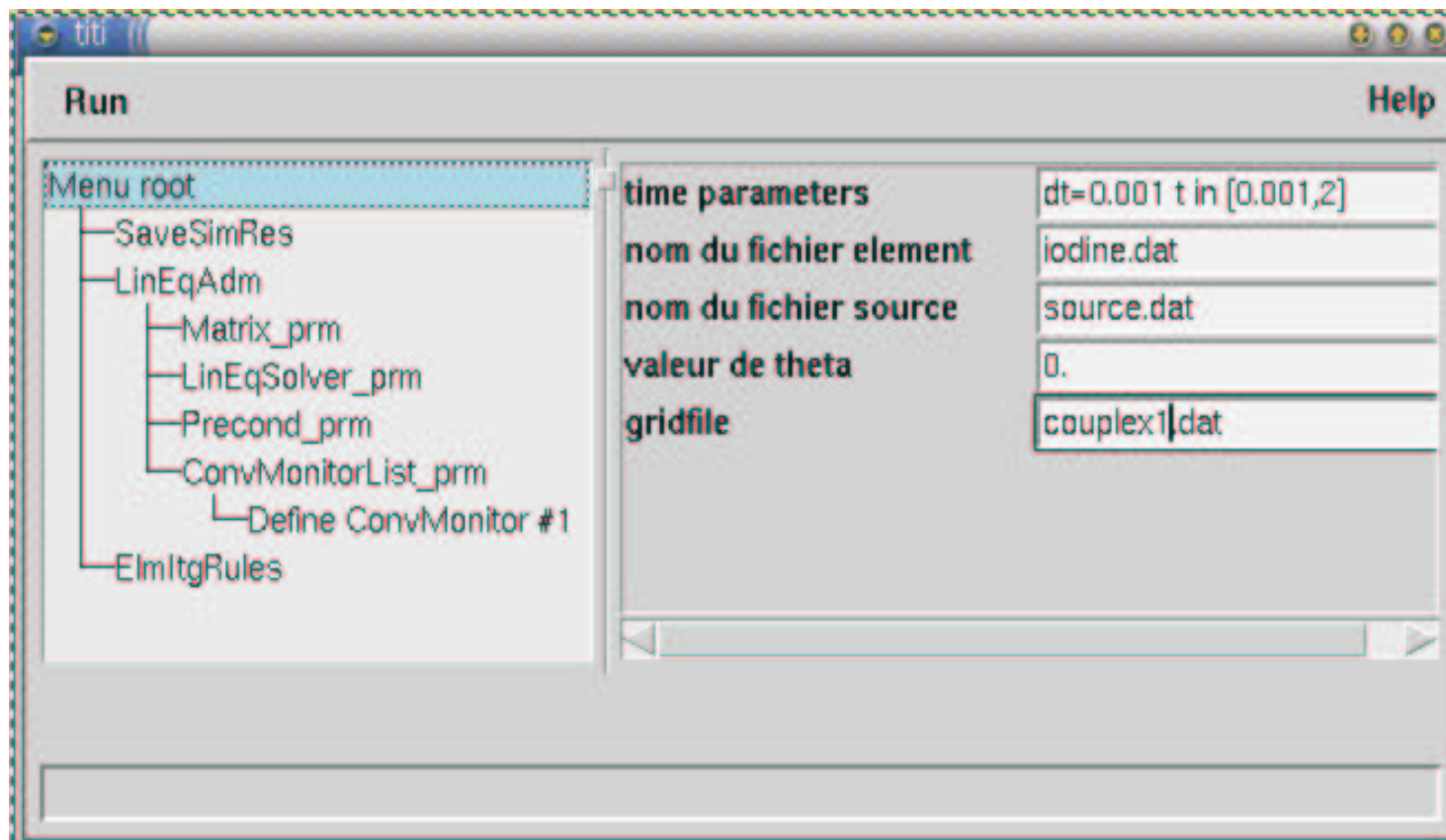
**Element types implemented in the class FiniteVolume :**



The parameters like :

nsd, the element type, the time parameters etc...

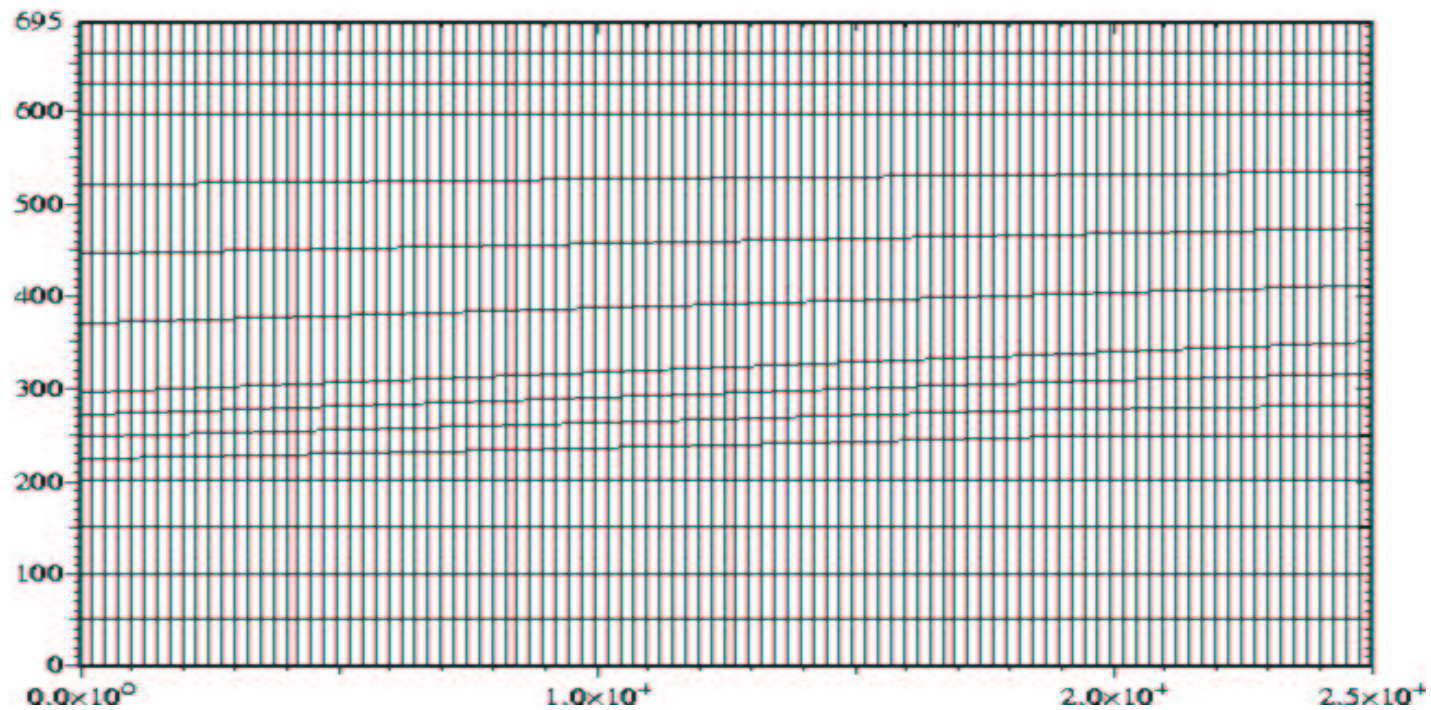
can be defined using a graphical interface :



# Numerical Results

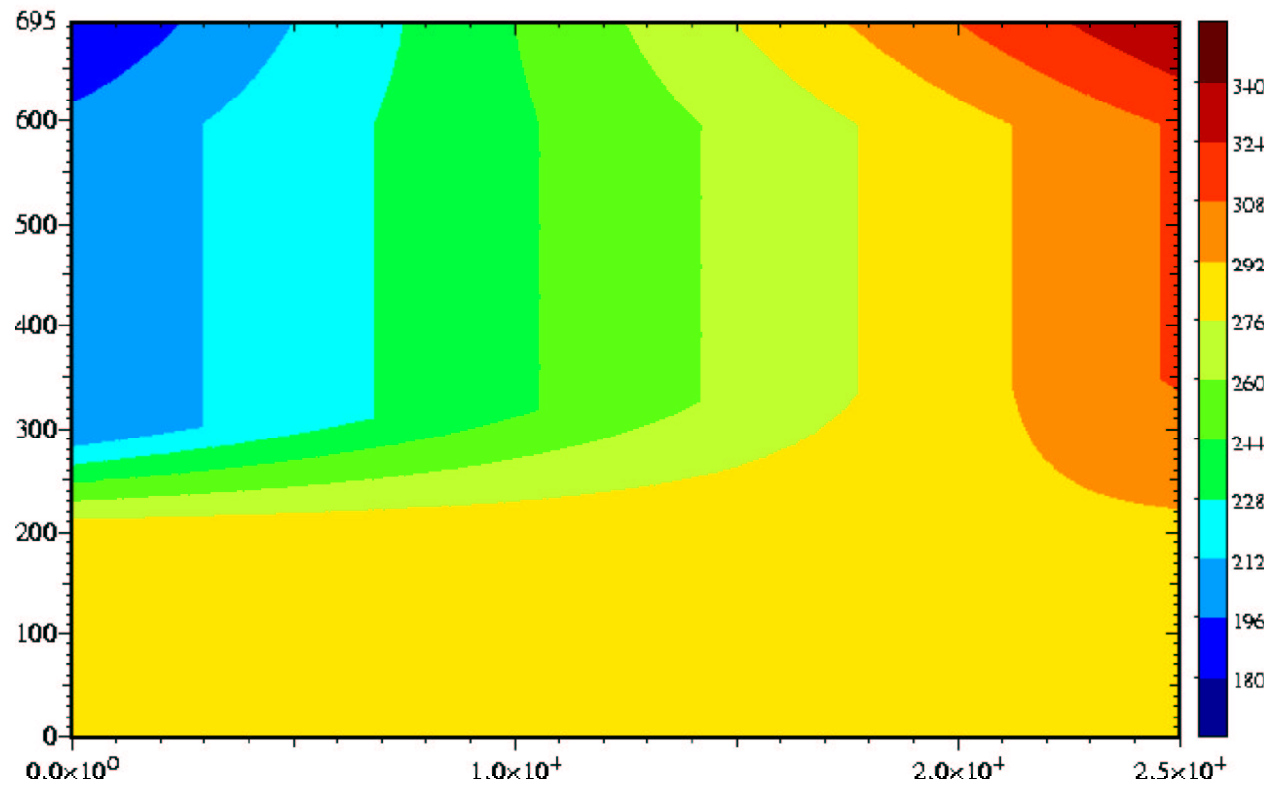
First mesh : 1500 elements, 1616 nodes

Mon Apr 2 10:01:24 2001



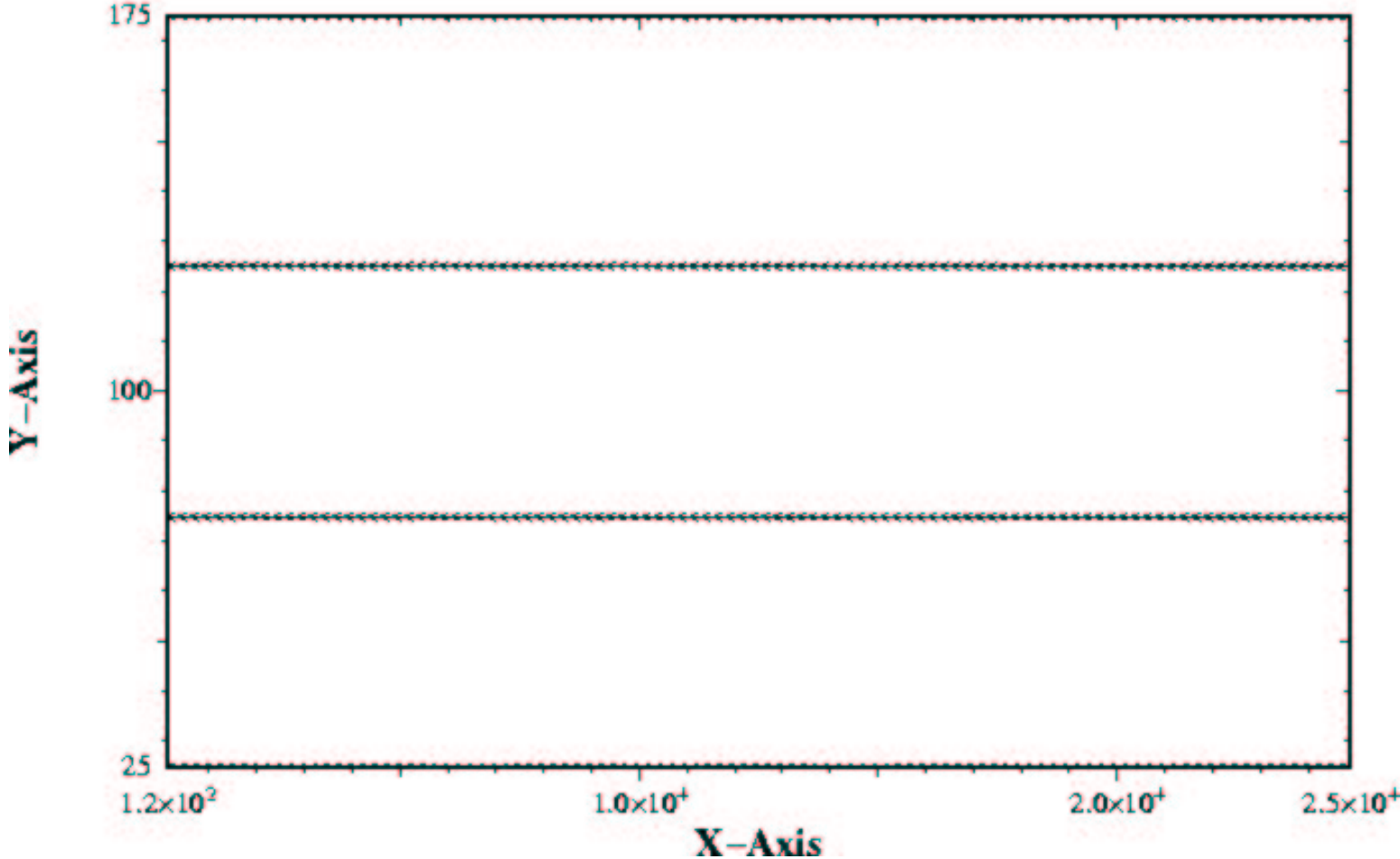
# The pressure

Pressure

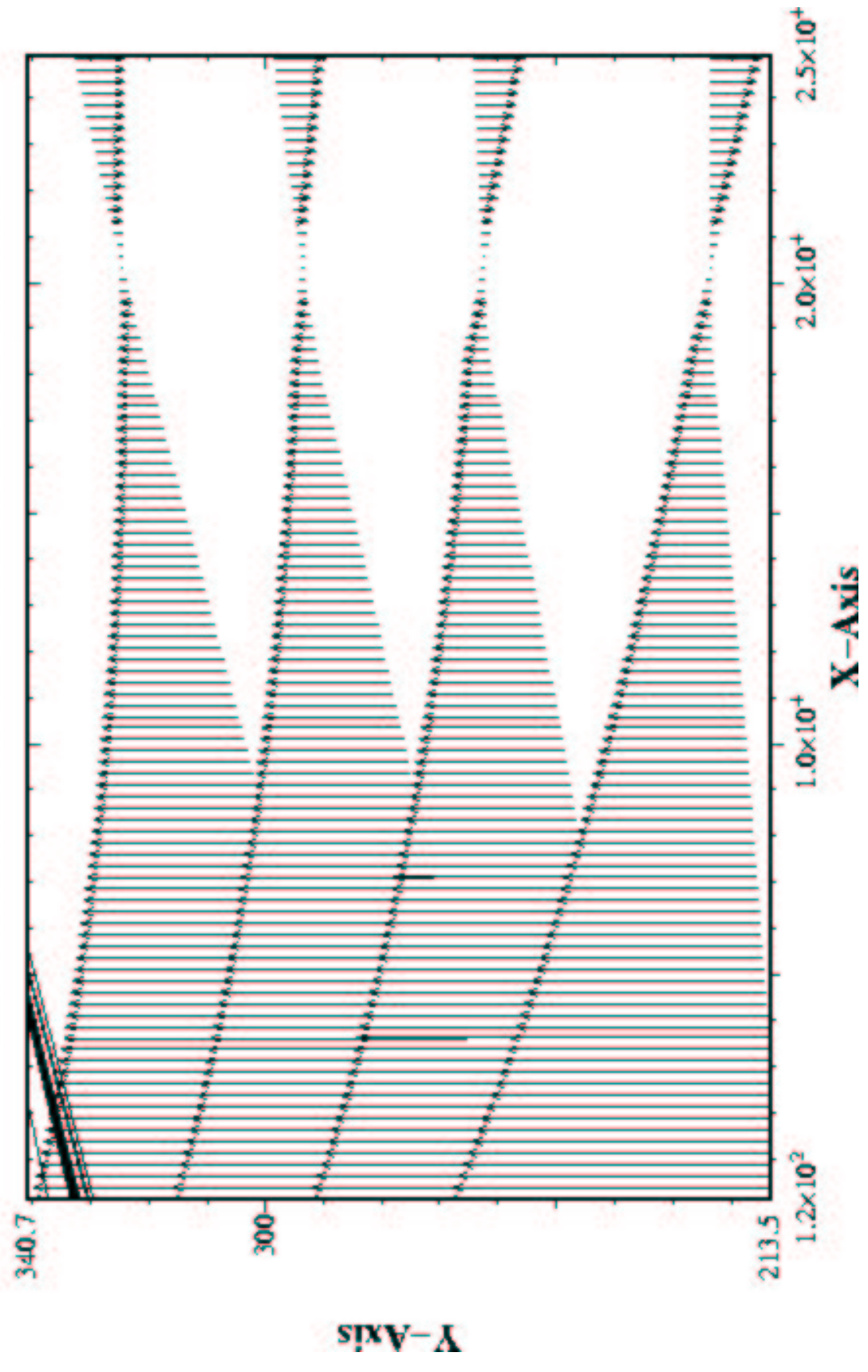


# Darcy Velocities

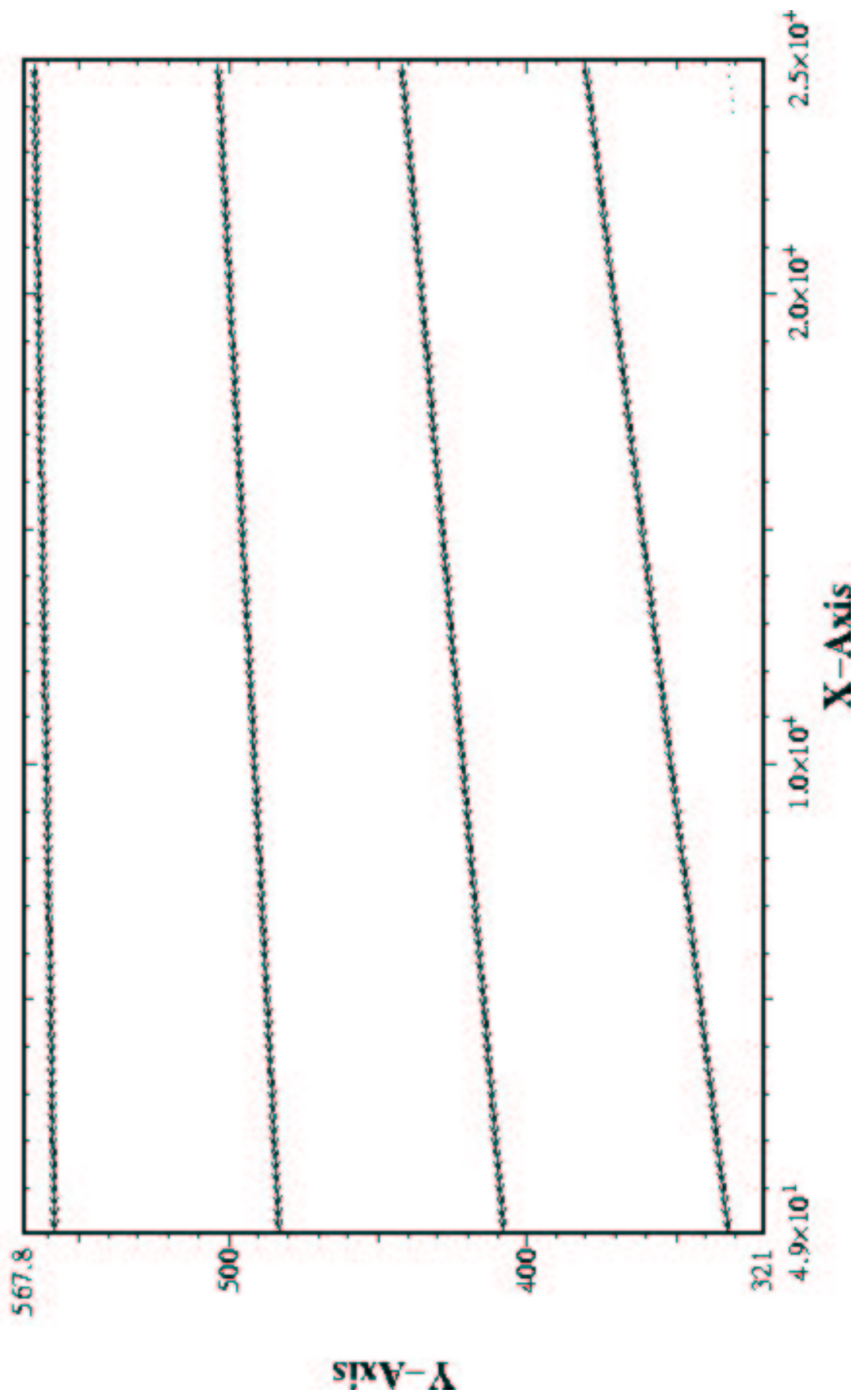
**Velocities for the Dogger Layer (scale=1.E+5)**



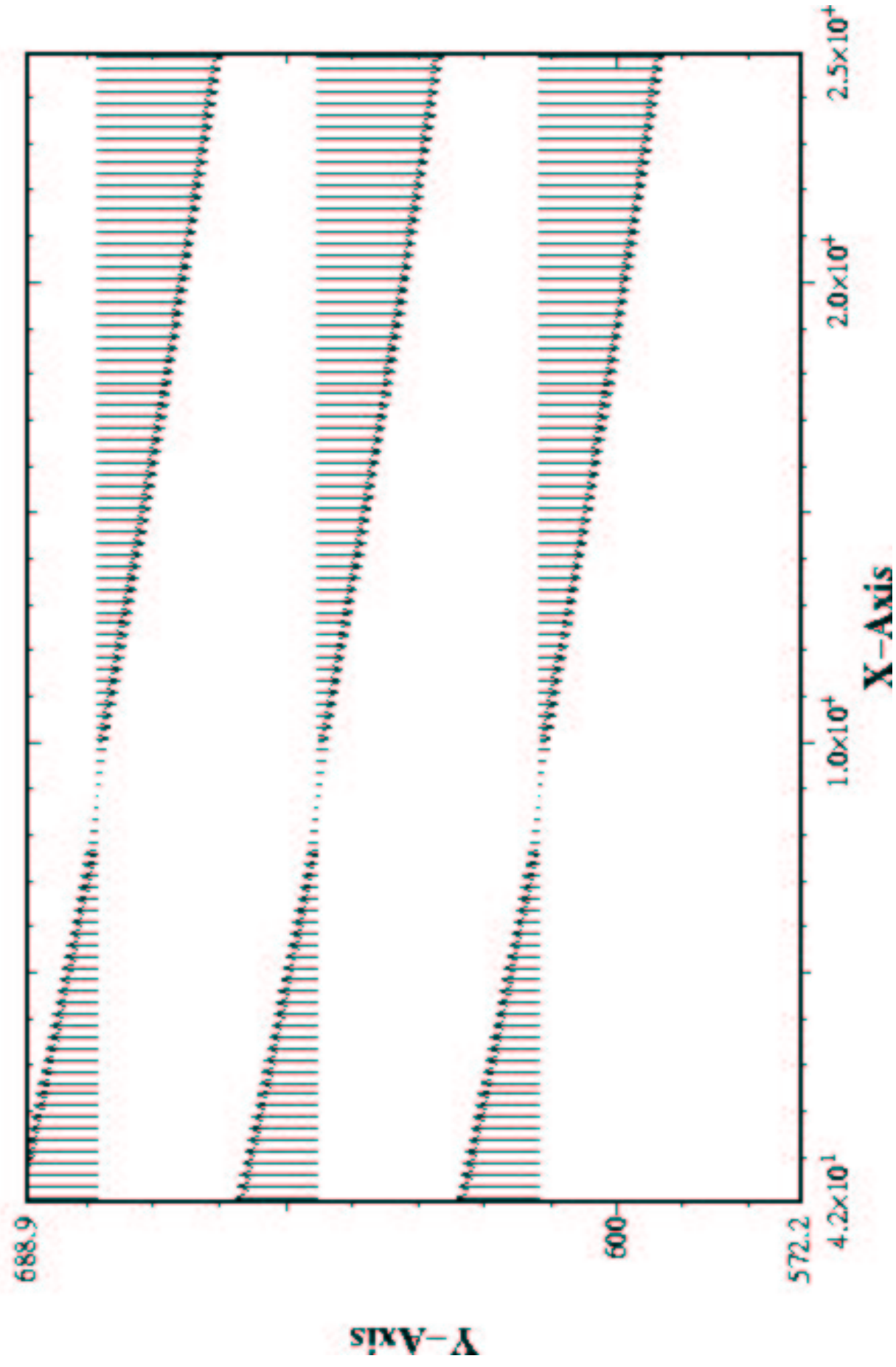
# Velocities for the Clay layer (scale=2.E+7)



# Velocities for the Limestone layer (scale=1.E+4)

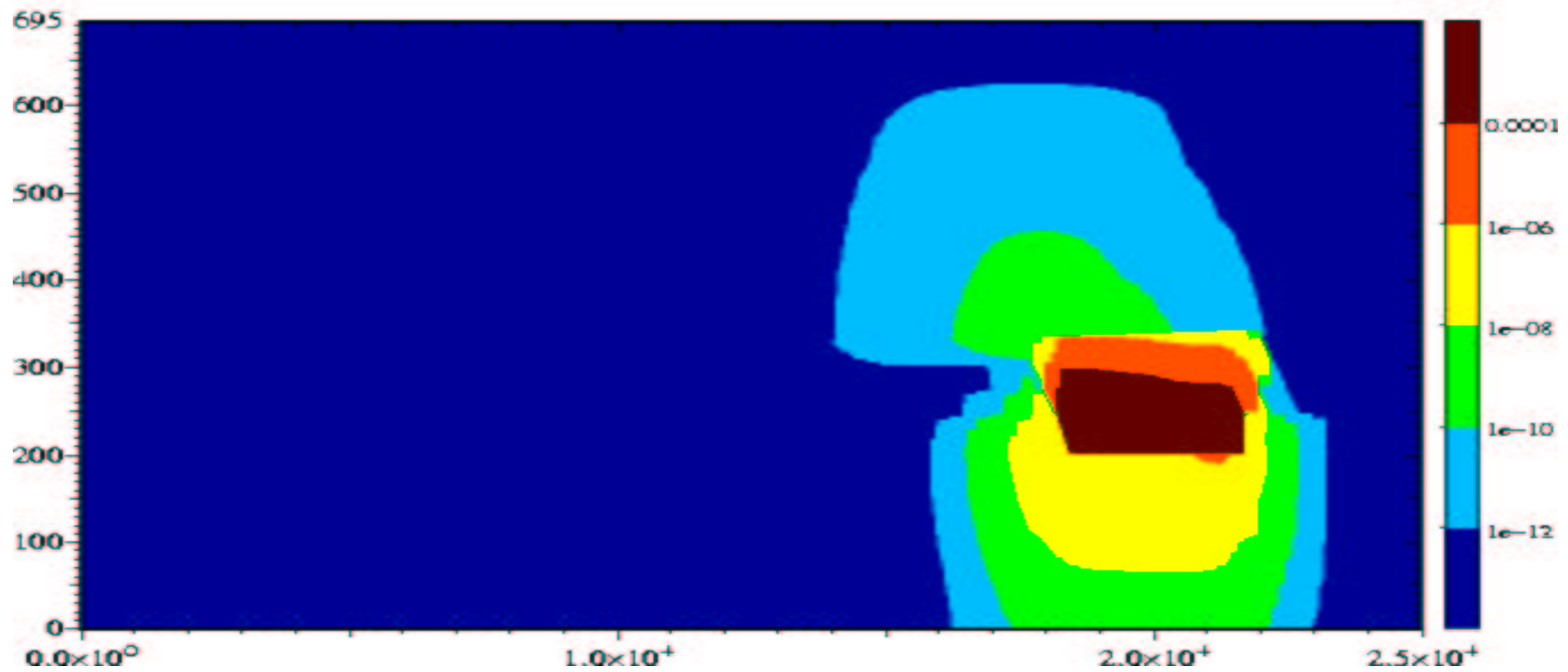


# Velocities for the Marl layer (scale=2.E+6)

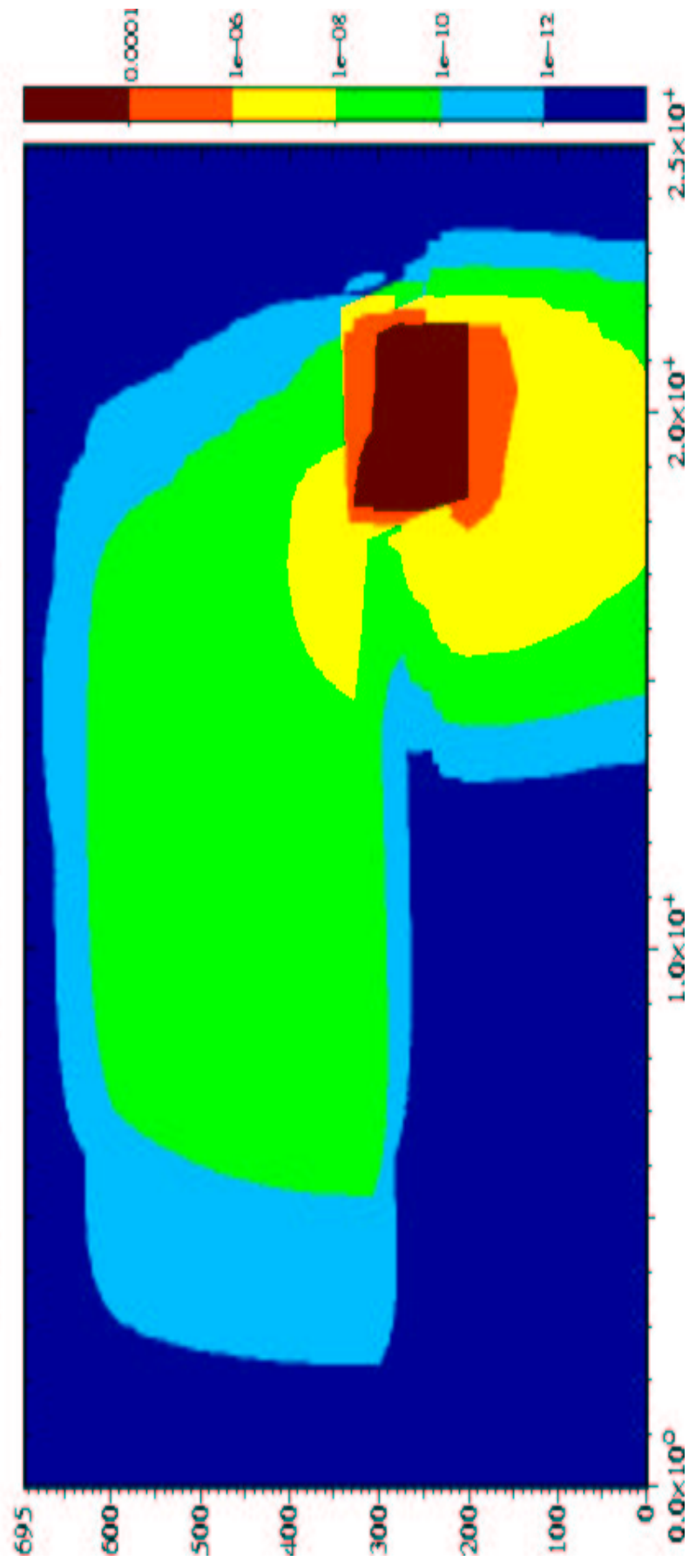


## Iodine concentration

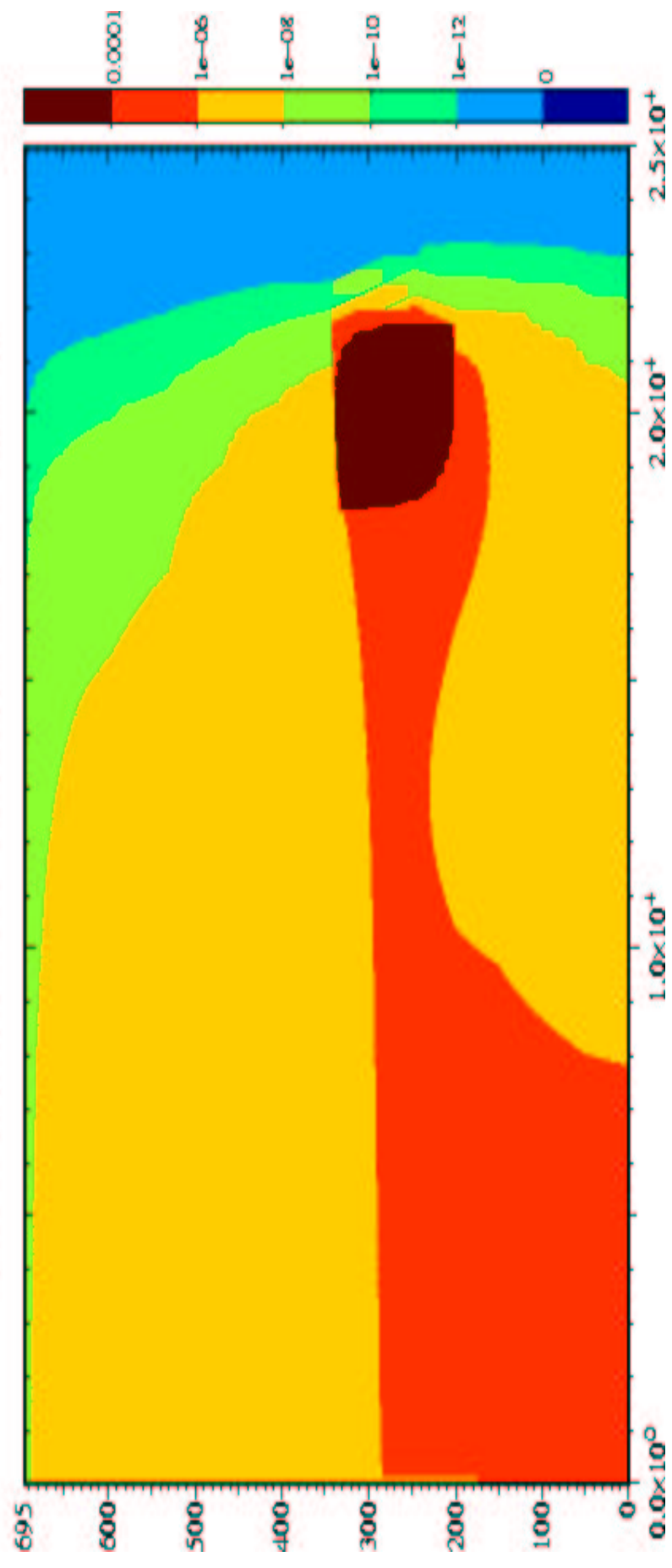
**Iodine concentration at time 10110 years**



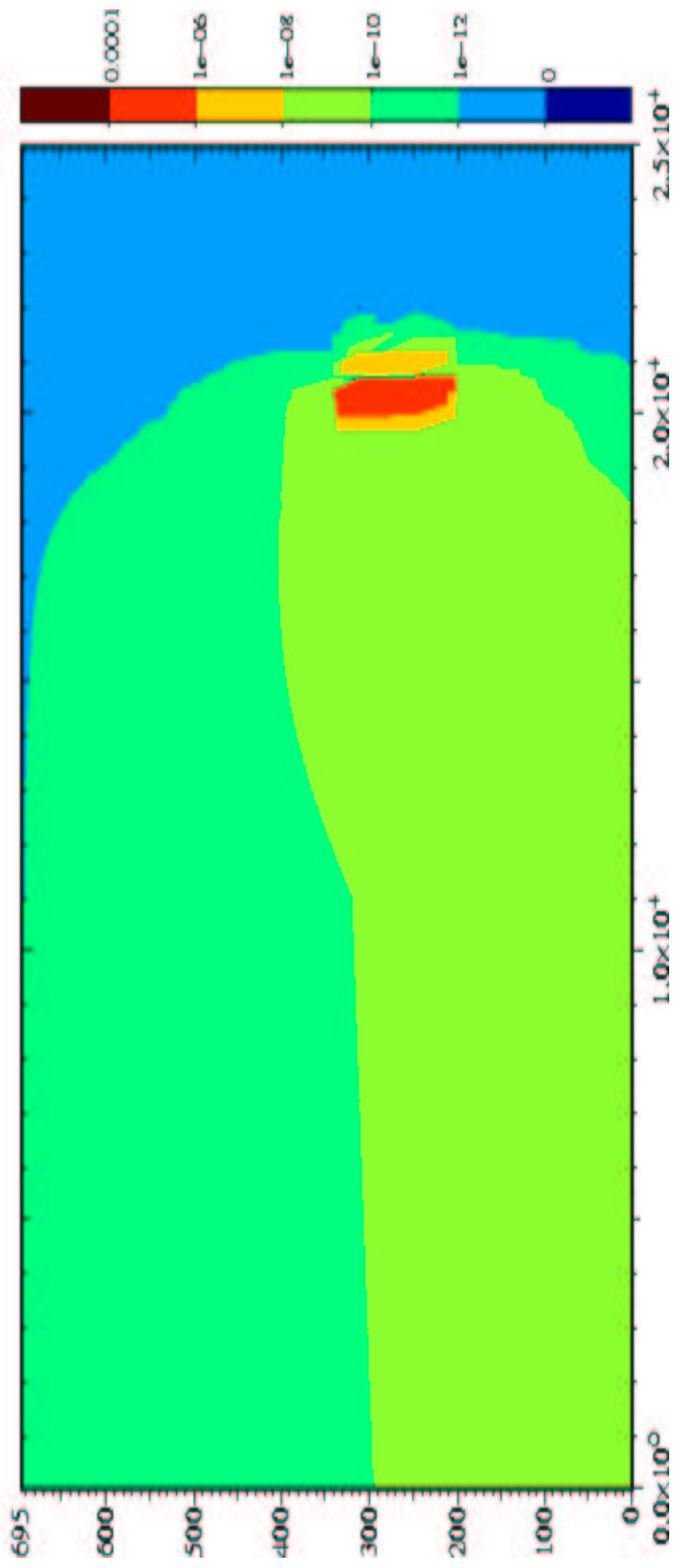
# Iodine Concentration at time 50110 years



**Iodine concentration at time 1E+6 years**

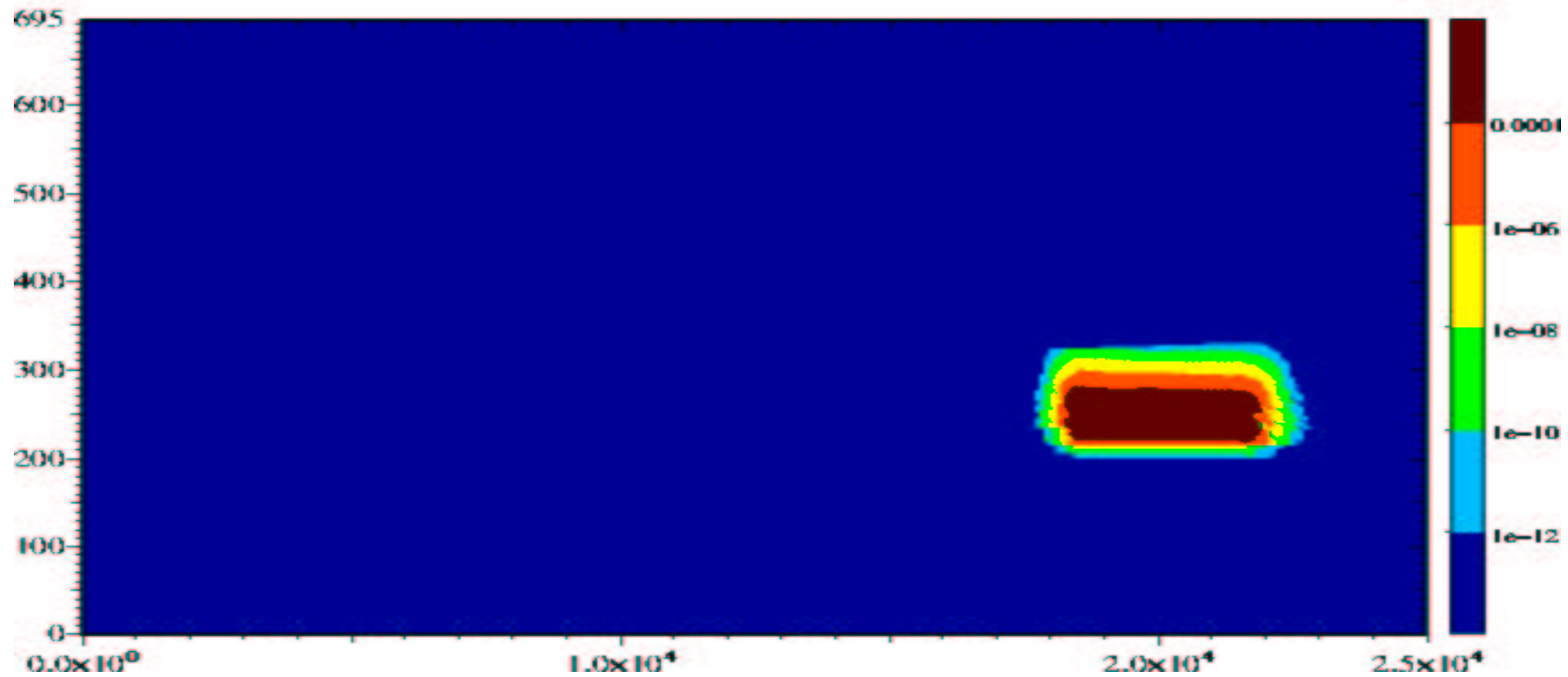


### Iodine Concentration at time 10.E7



Refined mesh : 13750 elements, 14076 nodes

### Iodine concentration at time 10110 Years



### Iodine concentration at time 50110 Years

