PANS modelling:

STD k-ε model

ε → equation:

$$\frac{\partial}{\partial t}(\alpha \rho \varepsilon) + \frac{\partial}{\partial x_i}(\alpha \rho u_i \varepsilon) - \frac{\partial}{\partial x_k}\left(\alpha \cdot \rho D_{\varepsilon eff} \frac{\partial \varepsilon}{\partial x_k}\right)$$

$$= C_1 \alpha \rho G \frac{\varepsilon}{k} - \left(\frac{2}{3} C_1 - C_3\right) \alpha \rho \frac{\partial u_i}{\partial x_i} \varepsilon - \left(C_2 \alpha \rho \frac{\varepsilon}{k}\right) \varepsilon + S_{\varepsilon_0} + S_{fvo\varepsilon}$$

$= 0 \to$ incompressible

$\to 0$ we are not specifying

kEpsilon.C

```
tmp<fvScalarMatrix> epsEqn
(
    fvm::ddt(alpha, rho, epsilon_)
  + fvm::div(alphaRhoPhi, epsilon_)
  - fvm::laplacian(alpha*rho*DepsilonEff(), epsilon_)
 ==
    C1_*alpha()*rho()*G*epsilon_()/k_()
  - fvm::SuSp(((2.0/3.0)*C1_ - C3_)*alpha()*rho()*divU, epsilon_)
  - fvm::Sp(C2_*alpha()*rho()*epsilon_()/k_(), epsilon_)
  + epsilonSource()
  + fvOptions(alpha, rho, epsilon_)
);
```

$$D_{\varepsilon eff} = \frac{\nu_t}{\sigma_\varepsilon} + \nu$$

kEpsilon.H

```
tmp<volScalarField> DepsilonEff() const
{
    return tmp<volScalarField>
    (
        new volScalarField
        (
            "DepsilonEff",
            (this->nut_/sigmaEps_ + this->nu())
        )
    );
}
```

$C_1, C_2 \to$ Model Constant

$\alpha = 1 \to$ single Phase

$\rho \Rightarrow$ constant.

k- Equation

kEpsilon.C

```
tmp<fvScalarMatrix> kEqn
(
    fvm::ddt(alpha, rho, k_)
  + fvm::div(alphaRhoPhi, k_)
  - fvm::laplacian(alpha*rho*DkEff(), k_)
 ==
    alpha()*rho()*G
  - fvm::SuSp((2.0/3.0)*alpha()*rho()*divU, k_)
  - fvm::Sp(alpha()*rho()*epsilon_()/k_(), k_)
  + kSource()
  + fvOptions(alpha, rho, k_)
);
```

$$\frac{\partial(\alpha\rho k)}{\partial t} + \frac{\partial}{\partial x_i}(\alpha\rho u_i k) - \frac{\partial}{\partial x_k}\left(\alpha\rho D_{keff}\frac{\partial k}{\partial x_k}\right)$$

$$= \alpha\rho G - \frac{2}{3}\alpha\rho\frac{\partial u_i}{\partial x_i}k$$

$$- \alpha\rho\varepsilon + S_k + S_{fvok}$$

kEpsilon.H

```
tmp<volScalarField> DkEff() const
{
    return tmp<volScalarField>
    (
        new volScalarField
        (
            "DkEff",
            (this->nut_/sigmak_ + this->nu())
        )
    );
}
```

$$D_{keff} = \frac{\nu_t}{\sigma_k} + \nu$$

KEPSILON.C

```
tmp<volTensorField> tgradU = fvc::grad(U);
volScalarField::Internal G
(
    this->GName(),
    nut.v()*(dev(twoSymm(tgradU().v())) && tgradU().v())
);
```

G = production of k =

$$= -\langle u_i' u_k' \rangle \frac{\partial \overline{U}_i}{\partial x_k}$$

Boussinesq approximation

$$-\langle u_i' u_k' \rangle = \nu_t\left(\frac{\partial \overline{U}_i}{\partial x_k} + \frac{\partial \overline{U}_k}{\partial x_i}\right) - \frac{2}{3}k\delta_{ij}$$

$$G = \nu_t * \left[\mathrm{dev}\left(\frac{\partial \overline{U}_i}{\partial x_k} + \frac{\partial \overline{U}_k}{\partial x_i}\right) : \left(\frac{\partial \overline{U}_i}{\partial x_k}\right)\right]$$

$$\mathrm{dev}(t) = t - \frac{1}{3}\,\mathrm{trace}(t) * I$$