The kOmegaSST model

The kOmegaSST model in OpenFOAM-1.6 (almost identical in newer)

From `$FOAM_SRC/turbulenceModels/incompressible/RAS/kOmegaSST/kOmegaSST.H`:

```
- Menter, F., Esch, T.
  "Elements of Industrial Heat Transfer Prediction"
  16th Brazilian Congress of Mechanical Engineering (COBEM),
  Nov. 2001
- Note that this implementation is written in terms of alpha diffusion
  coefficients rather than the more traditional sigma (alpha = 1/sigma)
  so that the blending can be applied to all coefficients in a
  consistent manner. The paper suggests that sigma is blended but this
  would not be consistent with the blending of the k-epsilon and
  k-omega models.
- Also note that the error in the last term of equation (2) relating to
  sigma has been corrected.
- Wall-functions are applied in this implementation by using equations
  (14) to specify the near-wall omega as appropriate.
- The blending functions (15) and (16) are not currently used because of
  the uncertainty in their origin, range of applicability and that is y+
  becomes sufficiently small blending u_tau in this manner clearly
  becomes nonsense.
```

$k-\omega$ SST: Equations

$$\frac{\partial k}{\partial t}+U_j\frac{\partial k}{\partial x_j}=P_k-\beta^* k\omega+\frac{\partial}{\partial x_j}\left[(\nu+\sigma_k\nu_t)\frac{\partial k}{\partial x_j}\right]$$

$$\frac{\partial \omega}{\partial t}+U_j\frac{\partial \omega}{\partial x_j}=\alpha S^2-\beta\omega^2+\frac{\partial}{\partial x_j}\left[(\nu+\sigma_\omega\nu_t)\frac{\partial \omega}{\partial x_j}\right]+2(1-F_1)\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i}$$

$$\nu_t=\frac{a_1 k}{\max(a_1\omega,SF_2)},\quad P_k=\min(G,10\beta^* k\omega),\quad G=\nu_t\frac{\partial U_i}{\partial x_j}\left(\frac{\partial U_i}{\partial x_j}+\frac{\partial U_j}{\partial x_i}\right)$$

$$S^2=\left|\frac{1}{2}(\partial_j u_i+\partial_i u_j)\right|^2,\quad S=\sqrt{S^2}=\left|\frac{1}{2}(\partial_j u_i+\partial_i u_j)\right|$$

$$F_1=\tanh\left\{\left\{\min\left(\min\left[\max\left(\frac{\sqrt{k}}{\beta^*\omega y},\frac{500\nu}{y^2\omega}\right),\frac{4\sigma_{\omega 2}k}{CD^+_{k\omega}y^2}\right],10\right)\right\}^4\right\}$$

$$F_2=\tanh\left[\left[\min\left(\max\left(\frac{2\sqrt{k}}{\beta^*\omega y},\frac{500\nu}{y^2\omega}\right),100\right)\right]^2\right]$$

$$CD_{k\omega}=2\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i}$$

Parameters that are blended using F_1:

$\sigma_k,\quad \sigma_\omega,\quad \alpha,\quad \beta$

Constants:

$\beta^*,\quad \sigma_{k1},\quad \sigma_{k2},\quad \sigma_{\omega 1},\quad \sigma_{\omega 2},$
$\alpha_1,\quad \alpha_2,\quad \beta_1,\quad \beta_2,\quad a_1$

$k-\omega$ SST in OpenFOAM-1.6, ν_t

Source code:

`$FOAM_SRC/turbulenceModels/incompressible/RAS/kOmegaSST`

Kinematic eddy viscosity:

$$\nu_t = \frac{a_1 k}{\max(a_1\omega, SF_2)}$$

kOmegaSST.C:

```
nut_ =
 a1_*k_/max(a1_*(omega_ + omegaSmall_), F2()*mag(symm(fvc::grad(U_))));
```

In `kOmegaSST.C`:

```
a1_(dimensioned<scalar>::lookupOrAddToDict("a1",coeffDict_,0.31))
```

In `kOmegaSST.C` (S = `sqrt(S2)`):

```
volScalarField S2 = magSqr(symm(fvc::grad(U_)));
```

i.e. $S^2 = \left|\frac{1}{2}(\partial_j u_i + \partial_i u_j)\right|^2$ and $S = \sqrt{S^2} = \left|\frac{1}{2}(\partial_j u_i + \partial_i u_j)\right|$

F2() is a blending function, which is described on the next slide

$k-\omega$ SST in OpenFOAM-1.6, F2()

F2() is a blending function:

$$F_2 = \tanh\left[\left[\min\left(\max\left(\frac{2\sqrt{k}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\right), 100\right)\right]^2\right]$$

In `kOmegaSST.C`:

```
tmp<volScalarField> kOmegaSST::F2() const
{
    volScalarField arg2 = min
    (
        max
        (
            (scalar(2)/betaStar_)*sqrt(k_)/(omega_*y_),
            scalar(500)*nu()/(sqr(y_)*omega_)
        ),
        scalar(100)
    );

    return tanh(sqr(arg2));
}
```

$k-\omega$ SST in OpenFOAM-1.6, Turbulence kinetic energy eq.

$$\frac{\partial k}{\partial t}+U_j\frac{\partial k}{\partial x_j}=P_k-\beta^* k\omega+\frac{\partial}{\partial x_j}\left[(\nu+\sigma_k\nu_t)\frac{\partial k}{\partial x_j}\right],\quad P_k=\min(G,10\beta^* k\omega),\quad G=\nu_t\frac{\partial U_i}{\partial x_j}\left(\frac{\partial U_i}{\partial x_j}+\frac{\partial U_j}{\partial x_i}\right)$$

In `kOmegaSST.C`:

```
tmp<fvScalarMatrix> kEqn
(
    fvm::ddt(k_)
  + fvm::div(phi_, k_)
  - fvm::Sp(fvc::div(phi_), k_)
  - fvm::laplacian(DkEff(F1), k_)
 ==
    min(G, c1_*betaStar_*k_*omega_)
  - fvm::Sp(betaStar_*omega_, k_)
);
```

The effective diffusivity for k, (`DkEff(F1)`), is described on a later slide.
F1 is obtained from F1(), which is a blending function for σ_k, and is described on the next slide,

where $CD_{k\omega}=2\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i}$

```
volScalarField CDkOmega =
    (2*alphaOmega2_)*(fvc::grad(k_) & fvc::grad(omega_))/omega_;
```

F1() is a blending function, `kOmegaSST.C` (compressed here):

$$F_1 = \tanh\left\{\left\{\min\left(\min\left[\max\left(\frac{\sqrt{k}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\right), \frac{4\sigma_{\omega 2}k}{CD^+_{k\omega}y^2}\right], 10\right)\right\}^4\right\}$$

```
tmp<volScalarField> kOmegaSST::F1(const volScalarField& CDkOmega) const
{   volScalarField CDkOmegaPlus = max
    (   CDkOmega,
        dimensionedScalar("1.0e-10", dimless/sqr(dimTime), 1.0e-10)
    );
    volScalarField arg1 = min
    ( min
        ( max
            (   (scalar(1)/betaStar_)*sqrt(k_)/(omega_*y_),
                scalar(500)*nu()/(sqr(y_)*omega_)
            ),
            (4*alphaOmega2_)*k_/(CDkOmegaPlus*sqr(y_))
        ),
        scalar(10)
    );
    return tanh(pow4(arg1));}
```

$F_1 = 0$ in the freestream ($k-\varepsilon$ model) and $F_1 = 1$ in the boundary layer ($k-\omega$ model)

$k-\omega$ SST in OpenFOAM-1.6, Effective diffusivity for k

The effective diffusivity for k, (`DkEff(F1)`), `kOmegaSST.H`:

```
tmp<volScalarField> DkEff(const volScalarField& F1) const
{
    return tmp<volScalarField>
    (
        new volScalarField("DkEff", alphaK(F1)*nut_ + nu())
    );
}
```

Blend `alphaK1` and `alphaK2` using blend function `F1`, `kOmegaSST.H`:

```
tmp<volScalarField> alphaK
(
    const volScalarField& F1
) const
{
    return blend(F1, alphaK1_, alphaK2_);
}
```

In `kOmegaSST.C`:

```
alphaK1_(dimensioned<scalar>::lookupOrAddToDict("alphaK1",coeffDict_,0.85034))
alphaK2_(dimensioned<scalar>::lookupOrAddToDict("alphaK2",coeffDict_,1.0))
```

$k-\omega$ SST in OpenFOAM-1.6, Specific dissipation rate eq.

$$\frac{\partial \omega}{\partial t} + U_j \frac{\partial \omega}{\partial x_j} = \alpha S^2 - \beta \omega^2 + \frac{\partial}{\partial x_j}\left[(\nu + \sigma_\omega \nu_t)\frac{\partial \omega}{\partial x_j}\right] + 2(1-F_1)\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i}$$

In `kOmegaSST.C`:

```
tmp<fvScalarMatrix> omegaEqn
(
    fvm::ddt(omega_)
  + fvm::div(phi_, omega_)
  - fvm::Sp(fvc::div(phi_), omega_)
  - fvm::laplacian(DomegaEff(F1), omega_)
 ==
    gamma(F1)*2*S2
  - fvm::Sp(beta(F1)*omega_, omega_)
  - fvm::SuSp
    (
        (F1 - scalar(1))*CDkOmega/omega_,
        omega_
    )
);
```