SfePy Documentation

October 23, 2008

Contents

1 Notation

Ω	volume (sub)domain
Γ	surface (sub)domain
continued...	

… *continued*	
t	time
y	any function
$\underline{y}$	any vector function
$\underline{n}$	unit outward normal
q, s	scalar test function
p, r	scalar unknown or parameter function
$\bar{p}$	scalar parameter function
$\underline{v}$	vector test function
$\underline{w}$, $\underline{u}$	vector unknown or parameter function
$\underline{b}$	vector parameter function
$\underline{\underline{e}}(\underline{u})$	Cauchy strain tensor ($\frac{1}{2}((\nabla u) + (\nabla u)^T)$)
$\underline{\underline{F}}$	deformation gradient$F_{ij} = \frac{\partial x_i}{\partial \partial X_j}$
J	$\det(F)$
$\underline{\underline{C}}$	right Cauchy-Green deformation tensor $C = F^T F$
$\underline{\underline{E}}(\underline{u})$	Green strain tensor $E_{ij} = \frac{1}{2}(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} + \frac{\partial u_m}{\partial x_i}\frac{\partial u_m}{\partial x_j})$
$\underline{\underline{S}}$	second Piola-Kirchhoff stress tensor
$\underline{f}$	vector volume forces
f	scalar volume force (source)
ρ	density
ν	kinematic viscosity
c	any constant
$\delta_{ij}, \underline{\underline{I}}$	Kronecker delta, identity matrix

The suffix "$_0$" denotes a quatity related to a previous time step.
Term names are prefixed according to the following conventions:

dw	discrete weak	terms having a virtual (test) argument and zero or more unknown arguments, used for FE assembling
d	discrete	terms having all arguments known, the result is the scalar value of the integral
di	discrete integrated	like 'd' but the result is not a scalar (e.g. a vector)
dq	discrete quadrature	terms having all arguments known, the result are the values in quadrature points of elements
de	discrete element	terms having all arguments known, the result is a vector of integral averages over elements (element average of 'dq')

2 List of all terms

section	name	definition
(5.5)	de_average_variable	vector of $\forall K \in \mathcal{T}_h : \int_{T_K} y / \int_{T_K} 1$
(14.1)	dw_biot	$\int_\Omega p\ \alpha_{ij} e_{ij}(\underline{v}),\ \int_\Omega q\ \alpha_{ij} e_{ij}(\underline{u})$
(14.2)	dw_biot_th	$\int_\Omega \left[\int_0^t \alpha_{ij}(t-\tau)\, p(\tau))\ \mathrm{d}\tau\right]\ e_{ij}(\underline{v})$, $\int_\Omega \left[\int_0^t \alpha_{ij}(t-\tau) e_{kl}(\underline{u}(\tau))\ \mathrm{d}\tau\right] q$
(13.1)	de_cauchy_strain	vector of $\forall K \in \mathcal{T}_h : \int_{T_K} \underline{\underline{e}}(\underline{w}) / \int_{T_K} 1$
(13.2)	de_cauchy_stress	vector of $\forall K \in \mathcal{T}_h : \int_{T_K} D_{ijkl} e_k l(\underline{w}) / \int_{T_K} 1$
(7.3)	dw_convect	$\int_\Omega ((\underline{u} \cdot \nabla)\underline{u}) \cdot \underline{v}$
(6.2)	dw_diffusion	$\int_\Omega K_{ij} \nabla_i q \nabla_j p,\ \int_\Omega K_{ij} \nabla_i \bar{p} \nabla_j r$
(6.1)	de_diffusion_velocity	vector of $\forall K \in \mathcal{T}_h : \int_{T_K} K_{ij} \nabla_j r / \int_{T_K} 1$
(7.4)	dw_div_grad	$\int_\Omega \nu\ \nabla \underline{v} : \nabla \underline{u}$
(7.1)	dq_grad	$(\nabla p)\|_{qp}$
(6.3)	dw_laplace	$c \int_\Omega \nabla s \cdot \nabla r$ or $\sum_{K \in \mathcal{T}_h} \int_{T_K} c_K\ \nabla s \cdot \nabla r$
(7.2)	dq_lin_convect	$((\underline{b} \cdot \nabla)\underline{u})\|_{qp}$
(7.5)	dw_lin_convect	$\int_\Omega ((\underline{b} \cdot \nabla)\underline{u}) \cdot \underline{v}$
(13.3)	dw_lin_elastic	$\int_\Omega D_{ijkl}\ e_{ij}(\underline{v}) e_{kl}(\underline{u})$
(13.4)	dw_lin_elastic_iso	$\int_\Omega D_{ijkl}\ e_{ij}(\underline{v}) e_{kl}(\underline{u})$ with $D_{ijkl} = \mu(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk}) + \lambda\ \delta_{ij}\delta_{kl}$
(13.5)	dw_lin_elastic_th	$\int_\Omega \left[\int_0^t \mathcal{H}_{ijkl}(t-\tau)\ \frac{\mathrm{d}e_{kl}(\underline{u}(\tau))}{\mathrm{d}\tau}\ \mathrm{d}\tau\right]\ e_{ij}(\underline{v})$
(4.1)	dw_mass	$\int_\Omega \rho \underline{v} \cdot \frac{\underline{u} - \underline{u}_0}{\Delta t}$
(4.2)	dw_mass_scalar	$\int_\Omega qp$
(4.3)	dw_mass_scalar_fine_coarse	$\int_\Omega q_h p_H$
(4.4)	dw_mass_scalar_variable	$\int_\Omega cqp$
(4.5)	dw_mass_vector	$\int_\Omega \rho\ \underline{v} \cdot \underline{u}$
(6.4)	dw_permeability_r	$\int_\Omega K_{ij} \nabla_j q$
(11.1)	dw_piezo_coupling	$\int_\Omega g_{kij}\ e_{ij}(\underline{u}) \nabla_k q,\ \int_\Omega g_{kij}\ e_{ij}(\underline{v}) \nabla_k p$
(9.1)	dw_point_lspring	$\underline{f}^i = -k\underline{u}^i \quad \forall$ FE node i in region
(7.6)	dw_st_grad_div	$\gamma \int_\Omega (\nabla \cdot \underline{u}) \cdot (\nabla \cdot \underline{v})$
(7.7)	dw_st_pspg_c	$\sum_{K \in \mathcal{T}_h} \int_{T_K} \tau_K\ ((\underline{b} \cdot \nabla)\underline{u}) \cdot \nabla q$
(7.8)	dw_st_pspg_p	$\sum_{K \in \mathcal{T}_h} \int_{T_K} \tau_K\ \nabla p \cdot \nabla q$
(7.9)	dw_st_supg_c	$\sum_{K \in \mathcal{T}_h} \int_{T_K} \delta_K\ ((\underline{b} \cdot \nabla)\underline{u}) \cdot ((\underline{b} \cdot \nabla)\underline{v})$
(7.10)	dw_st_supg_p	$\sum_{K \in \mathcal{T}_h} \int_{T_K} \delta_K\ \nabla p \cdot ((\underline{b} \cdot \nabla)\underline{v})$
(7.11)	dw_stokes	$\int_\Omega p\ \nabla \cdot \underline{v},\ \int_\Omega q\ \nabla \cdot \underline{u}$
(5.1)	d_surface_dot	$\int_\Gamma pr,\ \int_\Gamma \underline{u} \cdot \underline{w}$
(5.9)	dw_surface_integrate	$\int_\Gamma q$

continued...

… *continued*		
(5.2)	d_surface_integrate	$\int_\Gamma y$, for vectors: $\int_\Gamma \underline{y} \cdot \underline{n}$
(12.1)	dw_surface_ltr	$\int_\Gamma \underline{v} \cdot \underline{\underline{\sigma}} \cdot \underline{n}$
(8.1)	dw_tl_bulk_penalty	$\int_\Omega S_{ij}(\underline{u}) \delta E_{ij}(\underline{u}; \underline{v})$
(8.2)	dw_tl_he_mooney_rivlin	$\int_\Omega S_{ij}(\underline{u}) \delta E_{ij}(\underline{u}; \underline{v})$
(8.3)	dw_tl_he_neohook	$\int_\Omega S_{ij}(\underline{u}) \delta E_{ij}(\underline{u}; \underline{v})$
(5.3)	d_volume	$\int_\Omega 1$
(5.6)	de_volume_average_mat	$\forall K \in \mathcal{T}_h : \int_{T_K} m / \int_{T_K} 1$
(5.4)	d_volume_dot	$\int_\Omega pr$, $\int_\Omega \underline{u} \cdot \underline{w}$
(5.10)	dw_volume_integrate	$\int_\Omega q$
(5.7)	di_volume_integrate	$\int_\Omega y$, $\int_\Omega \underline{y}$
(5.8)	di_volume_integrate_mat	$\int_\Omega m$
(10.1)	dw_volume_lvf	$\int_\Omega \underline{f} \cdot \underline{v}$ or $\int_\Omega f q$
(5.11)	dw_volume_wdot	$\int_\Omega yqp$, $\int_\Omega y \underline{v} \cdot \underline{u}$, $\int_\Omega ypr$, $\int_\Omega y \underline{u} \cdot \underline{w}$
(5.12)	dw_volume_wdot_scalar_th	$\int_\Omega \left[\int_0^t \mathcal{G}(t-\tau) p(\tau) \, \mathrm{d}\tau \right] q$

3 Introduction

Equations in SfePy are built using terms, which correspond directly to the integral forms of weak formulation of a problem to be solved. As an example, let us consider the Laplace equation in time interval $t \in [0, t_{\rm final}]$:

$$\frac{\partial T}{\partial t} + c \Delta T = 0 \text{ in } \Omega, \quad T(t) = \bar{T}(t) \text{ on } \Gamma \,. \tag{1}$$

The weak formulation of (1) is: Find $T \in V$, such that

$$\int_\Omega s \frac{\partial T}{\partial t} + \int_\Omega c \, \nabla T : \nabla s = 0, \quad \forall s \in V_0 \,, \tag{2}$$

where we assume no fluxes over $\partial\Omega \setminus \Gamma$. In the syntax used in SfePy input files, this can be written as

```
dw_mass_scalar.i1.Omega( s, dT/dt ) + dw_laplace.i1.Omega( coef, s, T ) = 0 ,   (3)
```

which directly corresponds to the discrete version of (2): Find $\boldsymbol{T} \in V_h$, such that

$$\boldsymbol{s}^T (\int_{\Omega_h} \boldsymbol{\phi}^T \boldsymbol{\phi}) \frac{\partial \boldsymbol{T}}{\partial t} + \boldsymbol{s}^T (\int_{\Omega_h} c \, \boldsymbol{G}^T \boldsymbol{G}) \boldsymbol{T} = 0, \quad \forall \boldsymbol{s} \in V_{h0} \,,$$

where $u \approx \boldsymbol{\phi u}$, $\nabla u \approx \boldsymbol{G u}$ for $u \in \{s, T\}$. The integrals over the discrete domain Ω_h are approximated by a numerical quadrature, that is named `i1` in our case.

3.1 Term call syntax

In general, the syntax of a term call in SfePy is:

```
<term_name>.<i>.<r>( <arg1>, <arg2>, ... ),
```

where `<i>` denotes an integral name (i.e. a name of numerical quadrature to use) and `<r>` marks a region (domain of the integral). In the following, `<virtual>` corresponds to a test function, `<state>` to a unknown function and `<parameter>` to a known function arguments. We will now describe all the terms available in SfePy to date.

4 Terms in termsMass

4.1 dw_mass

Class: MassTerm
Description: Inertial forces term (constant density).
Definition:

$$\int_\Omega \rho \underline{v} \cdot \frac{\underline{u} - \underline{u}_0}{\Delta t}$$

Arguments:

material.rho	ρ
ts.dt	Δt
parameter	$\underline{u}_0$

Syntax: `dw_mass.<i>.<r>( <ts>, <material>, <virtual>, <state>, <parameter> )`

4.2 dw_mass_scalar

Class: MassScalarTerm
Description: Scalar field mass matrix/rezidual.
Definition:

$$\int_\Omega qp$$

Syntax: `dw_mass_scalar.<i>.<r>( <virtual>, <state> )`

4.3 dw_mass_scalar_fine_coarse

Class: MassScalarFineCoarseTerm
Description: Scalar field mass matrix/rezidual for coarse to fine grid interpolation. Field p_H belong to the coarse grid, test field q_h to the fine grid.
Definition:

$$\int_\Omega q_h p_H$$

Syntax: `dw_mass_scalar_fine_coarse.<i>.<r>( <virtual>, <state>, <iemaps>, <pbase> )`

4.4 dw_mass_scalar_variable

Class: MassScalarVariableTerm
Description: Scalar field mass matrix/rezidual with coefficient c defined in nodes.
Definition:

$$\int_\Omega cqp$$

Syntax: `dw_mass_scalar_variable.<i>.<r>( <material>, <virtual>, <state> )`

4.5 dw_mass_vector

Class: MassVectorTerm
Description: Vector field mass matrix/rezidual.
Definition:

$$\int_\Omega \rho \, \underline{v} \cdot \underline{u}$$

Syntax: `dw_mass_vector.<i>.<r>( <material>, <virtual>, <state> )`

5 Terms in termsBasic

5.1 d_surface_dot

Class: DotProductSurfaceTerm
Description: Surface $L^2(\Gamma)$ dot product for both scalar and vector fields.
Definition:

$$\int_\Gamma pr, \; \int_\Gamma \underline{u} \cdot \underline{w}$$

Syntax: `d_surface_dot.<i>.<r>( <parameter_1>, <parameter_2> )`

5.2 d_surface_integrate

Class: IntegrateSurfaceTerm
Definition:

$$\int_\Gamma y, \text{ for vectors: } \int_\Gamma \underline{y} \cdot \underline{n}$$

Syntax: `d_surface_integrate.<i>.<r>( <parameter> )`

5.3 d_volume

Class: VolumeTerm
Description: Volume of a domain. Uses approximation of the parameter variable.
Definition:

$$\int_\Omega 1$$

Syntax: `d_volume.<i>.<r>( <parameter> )`

5.4 d_volume_dot

Class: DotProductVolumeTerm
Description: Volume $L^2(\Omega)$ dot product for both scalar and vector fields.
Definition:

$$\int_\Omega pr, \; \int_\Omega \underline{u} \cdot \underline{w}$$

Syntax: `d_volume_dot.<i>.<r>( <parameter_1>, <parameter_2> )`

5.5 de_average_variable

Class: AverageVariableTerm
Description: Variable y averaged in elements.
Definition: vector of

$$\forall K \in \mathcal{T}_h : \int_{T_K} y / \int_{T_K} 1$$

Syntax: `de_average_variable.<i>.<r>( <parameter> )`

5.6 de_volume_average_mat

Class: AverageVolumeMatTerm
Description: Material parameter m averaged in elements. Uses approximation of y variable.
Definition:

$$\forall K \in \mathcal{T}_h : \int_{T_K} m / \int_{T_K} 1$$

Arguments:

material	m (can have up to two dimensions)
parameter	y
shape	shape of material parameter parameter
mode	'const' or 'vertex' or 'element_avg'

Syntax: `de_volume_average_mat.<i>.<r>( <material>, <parameter>, <shape>, <mode> )`

5.7 di_volume_integrate

Class: IntegrateVolumeTerm
Definition:

$$\int_\Omega y, \int_\Omega \underline{y}$$

Syntax: `di_volume_integrate.<i>.<r>( <parameter> )`

5.8 di_volume_integrate_mat

Class: IntegrateVolumeMatTerm
Description: Integrate material parameter m over a domain. Uses approximation of y variable.
Definition:

$$\int_\Omega m$$

Arguments:

material	m (can have up to two dimensions)
parameter	y
shape	shape of material parameter parameter
mode	'const' or 'vertex' or 'element_avg'

Syntax: `di_volume_integrate_mat.<i>.<r>( <material>, <parameter>, <shape>, <mode> )`

5.9 dw_surface_integrate

Class: IntegrateSurfaceOperatorTerm
Definition:

$$\int_\Gamma q$$

Syntax: `dw_surface_integrate.<i>.<r>( <material>, <virtual> )`

5.10 dw_volume_integrate

Class: IntegrateVolumeOperatorTerm
Definition:

$$\int_\Omega q$$

Syntax: `dw_volume_integrate.<i>.<r>( <virtual> )`

5.11 dw_volume_wdot

Class: WDotProductVolumeTerm
Description: Volume $L^2(\Omega)$ weighted dot product for both scalar and vector (not implemented in weak form!) fields. Can be evaluated. Can use derivatives.
Definition:

$$\int_\Omega yqp,\ \int_\Omega y\underline{v}\cdot\underline{u},\ \int_\Omega ypr,\ \int_\Omega y\underline{u}\cdot\underline{w}$$

Arguments:

material	weight function y

Syntax: `dw_volume_wdot.<i>.<r>( <arguments> )` where `<arguments>` is one of:

```
<material>, <virtual>, <state>
<material>, <parameter_1>, <parameter_2>
```

5.12 dw_volume_wdot_scalar_th

Class: WDotSProductVolumeOperatorTHTerm
Description: Fading memory volume $L^2(\Omega)$ weighted dot product for scalar fields. Can use derivatives.
Definition:

$$\int_\Omega \left[\int_0^t \mathcal{G}(t-\tau)p(\tau)\ \mathrm{d}\tau\right] q$$

Syntax: `dw_volume_wdot_scalar_th.<i>.<r>( <ts>, <material>, <virtual>, <state> )`

6 Terms in termsLaplace

6.1 de_diffusion_velocity

Class: DiffusionVelocityTerm
Description: Diffusion velocity averaged in elements.
Definition: vector of

$$\forall K \in \mathcal{T}_h : \int_{T_K} K_{ij} \nabla_j r / \int_{T_K} 1$$

Syntax: de_diffusion_velocity.<i>.<r>(<material>, <parameter>)

6.2 dw_diffusion

Class: DiffusionTerm
Description: General diffusion term with permeability K_{ij} constant or given in mesh vertices. Can be evaluated. Can use derivatives.
Definition:

$$\int_\Omega K_{ij} \nabla_i q \nabla_j p,\ \int_\Omega K_{ij} \nabla_i \bar{p} \nabla_j r$$

Syntax: dw_diffusion.<i>.<r>(<arguments>) where <arguments> is one of:

<material>, <virtual>, <state>
<material>, <parameter_1>, <parameter_2>

6.3 dw_laplace

Class: LaplaceTerm
Description: Laplace term with c constant or constant per element.
Definition:

$$c \int_\Omega \nabla s \cdot \nabla r \text{ or } \sum_{K \in \mathcal{T}_h} \int_{T_K} c_K\ \nabla s \cdot \nabla r$$

Syntax: dw_laplace.<i>.<r>(<material>, <virtual>, <state>)

6.4 dw_permeability_r

Class: PermeabilityRTerm
Description: Special-purpose diffusion-like term with permeability K_{ij} constant or given in mesh vertices (to use on a right-hand side).
Definition:

$$\int_\Omega K_{ij} \nabla_j q$$

Syntax: dw_permeability_r.<i>.<r>(<material>, <virtual>, <index>)

7 Terms in termsNavierStokes

7.1 dq_grad

Class: GradQTerm
Description: Gradient term (weak form) in quadrature points.
Definition:

$$(\nabla p)|_{qp}$$

Syntax: dq_grad.<i>.<r>(<state>)

7.2 dq_lin_convect

Class: LinearConvectQTerm
Description: Linearized convective term evaluated in quadrature points.
Definition:

$$((\underline{b} \cdot \nabla) \underline{u})|_{qp}$$

Syntax: dq_lin_convect.<i>.<r>(<parameter>, <state>)

7.3 dw_convect

Class: ConvectTerm
Description: Nonlinear convective term.
Definition:

$$\int_{\Omega} ((\underline{u} \cdot \nabla) \underline{u}) \cdot \underline{v}$$

Syntax: `dw_convect.<i>.<r>( <virtual>, <state> )`

7.4 dw_div_grad

Class: DivGradTerm
Description: Diffusion term.
Definition:

$$\int_{\Omega} \nu \ \nabla \underline{v} : \nabla \underline{u}$$

Syntax: `dw_div_grad.<i>.<r>( <material>, <virtual>, <state> )`

7.5 dw_lin_convect

Class: LinearConvectTerm
Description: Linearized convective term.
Definition:

$$\int_{\Omega} ((\underline{b} \cdot \nabla) \underline{u}) \cdot \underline{v}$$

Syntax: `dw_lin_convect.<i>.<r>( <virtual>, <parameter>, <state> )`

7.6 dw_st_grad_div

Class: GradDivStabilizationTerm
Description: Grad-div stabilization term (γ is a global stabilization parameter).
Definition:

$$\gamma \int_{\Omega} (\nabla \cdot \underline{u}) \cdot (\nabla \cdot \underline{v})$$

Syntax: `dw_st_grad_div.<i>.<r>( <material>, <virtual>, <state> )`

7.7 dw_st_pspg_c

Class: PSPGCStabilizationTerm
Description: PSPG stabilization term, convective part (τ is a local stabilization parameter).
Definition:

$$\sum_{K \in \mathcal{T}_h} \int_{T_K} \tau_K \ ((\underline{b} \cdot \nabla) \underline{u}) \cdot \nabla q$$

Syntax: `dw_st_pspg_c.<i>.<r>( <material>, <virtual>, <parameter>, <state> )`

7.8 dw_st_pspg_p

Class: PSPGPStabilizationTerm
Description: PSPG stabilization term, pressure part (τ is a local stabilization parameter), alias to Laplace term dw_laplace.
Definition:

$$\sum_{K \in \mathcal{T}_h} \int_{T_K} \tau_K \ \nabla p \cdot \nabla q$$

Syntax: `dw_st_pspg_p.<i>.<r>( <material>, <virtual>, <state> )`

7.9 dw_st_supg_c

Class: SUPGCStabilizationTerm
Description: SUPG stabilization term, convective part (δ is a local stabilization parameter).
Definition:

$$\sum_{K \in \mathcal{T}_h} \int_{T_K} \delta_K \ ((\underline{b} \cdot \nabla) \underline{u}) \cdot ((\underline{b} \cdot \nabla) \underline{v})$$

Syntax: dw_st_supg_c.<i>.<r>(<material>, <virtual>, <parameter>, <state>)

7.10 dw_st_supg_p

Class: SUPGPStabilizationTerm
Description: SUPG stabilization term, pressure part (δ is a local stabilization parameter).
Definition:

$$\sum_{K \in \mathcal{T}_h} \int_{T_K} \delta_K \ \nabla p \cdot ((\underline{b} \cdot \nabla) \underline{v})$$

Syntax: dw_st_supg_p.<i>.<r>(<material>, <virtual>, <parameter>, <state>)

7.11 dw_stokes

Class: StokesTerm
Description: Stokes problem coupling term. Corresponds to weak forms of gradient and divergence terms. Can be evaluated.
Definition:

$$\int_\Omega p \ \nabla \cdot \underline{v}, \ \int_\Omega q \ \nabla \cdot \underline{u}$$

Syntax: dw_stokes.<i>.<r>(<arguments>) where <arguments> is one of:

```
<virtual>, <state>
<state>, <virtual>
<parameter_s>, <parameter_v>
```

8 Terms in termsHyperElasticity

8.1 dw_tl_bulk_penalty

Class: BulkPenaltyTerm
Description: Hyperelastic bulk penalty term. Stress $S_{ij} = K(J-1)\ JC_{ij}^{-1}$.
Definition:

$$\int_\Omega S_{ij}(\underline{u}) \delta E_{ij}(\underline{u}; \underline{v})$$

Syntax: dw_tl_bulk_penalty.<i>.<r>(<material>, <virtual>, <state>)

8.2 dw_tl_he_mooney_rivlin

Class: MooneyRivlinTerm
Description: Hyperelastic Mooney-Rivlin term. Effective stress $S_{ij} = \kappa J^{-\frac{4}{3}}(C_{kk}\delta_{ij} - C_{ij} - \frac{2}{3} I_2 C_{ij}^{-1})$.
Definition:

$$\int_\Omega S_{ij}(\underline{u}) \delta E_{ij}(\underline{u}; \underline{v})$$

Syntax: dw_tl_he_mooney_rivlin.<i>.<r>(<material>, <virtual>, <state>)

8.3 dw_tl_he_neohook

Class: NeoHookeanTerm
Description: Hyperelastic neo-Hookean term. Effective stress $S_{ij} = \mu J^{-\frac{2}{3}}(\delta_{ij} - \frac{1}{3}C_{kk}C_{ij}^{-1})$.
Definition:

$$\int_\Omega S_{ij}(\underline{u})\delta E_{ij}(\underline{u};\underline{v})$$

Syntax: `dw_tl_he_neohook.<i>.<r>( <material>, <virtual>, <state> )`

9 Terms in termsPoint

9.1 dw_point_lspring

Class: LinearPointSpringTerm
Description: Linear springs constraining movement of FE nodes in a reagion; use as a relaxed Dirichlet boundary conditions.
Definition:

$$\underline{f}^i = -k\underline{u}^i \quad \forall \text{ FE node } i \text{ in region}$$

Syntax: `dw_point_lspring.<i>.<r>( <material>, <virtual>, <state> )`

10 Terms in termsVolume

10.1 dw_volume_lvf

Class: LinearVolumeForceTerm
Description: Vector or scalar linear volume forces (weak form) — a right-hand side source term.
Definition:

$$\int_\Omega \underline{f} \cdot \underline{v} \text{ or } \int_\Omega f q$$

Syntax: `dw_volume_lvf.<i>.<r>( <material>, <virtual> )`

11 Terms in termsPiezo

11.1 dw_piezo_coupling

Class: PiezoCouplingTerm
Description: Piezoelectric coupling term.
Definition:

$$\int_\Omega g_{kij}\ e_{ij}(\underline{u})\nabla_k q,\ \int_\Omega g_{kij}\ e_{ij}(\underline{v})\nabla_k p$$

Syntax: `dw_piezo_coupling.<i>.<r>( <arguments> )` where `<arguments>` is one of:

```
<material>, <virtual>, <state>
<material>, <state>, <virtual>
```

12 Terms in termsSurface

12.1 dw_surface_ltr

Class: LinearTractionTerm
Description: Linear traction forces (weak form), where, depending on dimension of 'material' argument, $\underline{\underline{\sigma}} \cdot \underline{n}$ is $\bar{p}\underline{\underline{I}} \cdot \underline{n}$ for a given scalar pressure, $\underline{f}$ for a traction vector, and itself for a stress tensor.
Definition:

$$\int_{\Gamma} \underline{v} \cdot \underline{\underline{\sigma}} \cdot \underline{n}$$

Syntax: `dw_surface_ltr.<i>.<r>( <material>, <virtual> )`

13 Terms in termsLinElasticity

13.1 de_cauchy_strain

Class: CauchyStrainTerm
Description: Cauchy strain tensor averaged in elements.
Definition: vector of

$$\forall K \in \mathcal{T}_h : \int_{T_K} \underline{\underline{e}}(\underline{w}) / \int_{T_K} 1$$

Syntax: `de_cauchy_strain.<i>.<r>( <parameter> )`

13.2 de_cauchy_stress

Class: CauchyStressTerm
Description: Cauchy stress tensor averaged in elements.
Definition: vector of

$$\forall K \in \mathcal{T}_h : \int_{T_K} D_{ijkl} e_k l(\underline{w}) / \int_{T_K} 1$$

Syntax: `de_cauchy_stress.<i>.<r>( <material>, <parameter> )`

13.3 dw_lin_elastic

Class: LinearElasticTerm
Description: General linear elasticity term, with D_{ijkl} given in the usual matrix form exploiting symmetry: in 3D it is 6×6 with the indices ordered as $[11, 22, 33, 12, 13, 23]$, in 2D it is 3×3 with the indices ordered as $[11, 22, 12]$. Can be evaluated. Can use derivatives.
Definition:

$$\int_{\Omega} D_{ijkl}\ e_{ij}(\underline{v}) e_{kl}(\underline{u})$$

Syntax: `dw_lin_elastic.<i>.<r>( <arguments> )` where `<arguments>` is one of:

```
<material>, <virtual>, <state>
<material>, <parameter_1>, <parameter_2>
```

13.4 dw_lin_elastic_iso

Class: LinearElasticIsotropicTerm
Description: Isotropic linear elasticity term.
Definition:

$$\int_{\Omega} D_{ijkl}\ e_{ij}(\underline{v}) e_{kl}(\underline{u}) \text{ with } D_{ijkl} = \mu(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk}) + \lambda\ \delta_{ij}\delta_{kl}$$

Syntax: `dw_lin_elastic_iso.<i>.<r>( <material>, <virtual>, <state> )`

13.5 dw_lin_elastic_th

Class: LinearElasticTHTerm
Definition:

$$\int_\Omega \left[\int_0^t \mathcal{H}_{ijkl}(t-\tau)\,\frac{\mathrm{d}e_{kl}(\underline{u}(\tau))}{\mathrm{d}\tau}\,\mathrm{d}\tau\right] e_{ij}(\underline{v})$$

Syntax: dw_lin_elastic_th.<i>.<r>(<ts>, <material>, <virtual>, <state>)

14 Terms in termsBiot

14.1 dw_biot

Class: BiotTerm
Description: Biot coupling term with α_{ij} given in vector form exploiting symmetry: in 3D it has the indices ordered as $[11, 22, 33, 12, 13, 23]$, in 2D it has the indices ordered as $[11, 22, 12]$. Corresponds to weak forms of Biot gradient and divergence terms. Can be evaluated.
Definition:

$$\int_\Omega p\ \alpha_{ij} e_{ij}(\underline{v}),\ \int_\Omega q\ \alpha_{ij} e_{ij}(\underline{u})$$

Syntax: dw_biot.<i>.<r>(<arguments>) where <arguments> is one of:

```
<material>, <virtual>, <state>
<material>, <state>, <virtual>
<material>, <parameter_s>, <parameter_v>
```

14.2 dw_biot_th

Class: BiotTHTerm
Description: Can have time derivatives.
Definition:

$$\int_\Omega \left[\int_0^t \alpha_{ij}(t-\tau)\,p(\tau))\ \mathrm{d}\tau\right] e_{ij}(\underline{v}),\ \int_\Omega \left[\int_0^t \alpha_{ij}(t-\tau) e_{kl}(\underline{u}(\tau))\ \mathrm{d}\tau\right] q$$

Syntax: dw_biot_th.<i>.<r>(<arguments>) where <arguments> is one of:

```
<ts>, <material>, <virtual>, <state>
<ts>, <material>, <state>, <virtual>
```

15 Term caches in cachesFiniteStrain

15.1 finite_strain_tl

Class: FiniteStrainTLDataCache

```
cache = term.get_cache( 'finite_strain_tl', <index> )
data = cache( <data name>, <ig>, <ih>, state )
```

16 Term caches in cachesBasic

16.1 cauchy_strain

Class: CauchyStrainDataCache

```
cache = term.get_cache( 'cauchy_strain', <index> )
data = cache( <data name>, <ig>, <ih>, state, get_vector )
```

16.2 div_vector

Class: DivVectorDataCache

```
cache = term.get_cache( 'div_vector', <index> )
data = cache( <data name>, <ig>, <ih>, state )
```

16.3 grad_scalar

Class: GradScalarDataCache

```
cache = term.get_cache( 'grad_scalar', <index> )
data = cache( <data name>, <ig>, <ih>, state )
```

16.4 grad_vector

Class: GradVectorDataCache

```
cache = term.get_cache( 'grad_vector', <index> )
data = cache( <data name>, <ig>, <ih>, state )
```

16.5 mat_in_qp

Class: MatInQPDataCache

```
cache = term.get_cache( 'mat_in_qp', <index> )
data = cache( <data name>, <ig>, <ih>, mat, ap, assumed_shapes, mode_in )
```

16.6 state_in_surface_qp

Class: StateInSurfaceQPDataCache

```
cache = term.get_cache( 'state_in_surface_qp', <index> )
data = cache( <data name>, <ig>, <ih>, state )
```

16.7 state_in_volume_qp

Class: StateInVolumeQPDataCache

```
cache = term.get_cache( 'state_in_volume_qp', <index> )
data = cache( <data name>, <ig>, <ih>, state, get_vector )
```

16.8 volume

Class: VolumeDataCache

```
cache = term.get_cache( 'volume', <index> )
data = cache( <data name>, <ig>, <ih>, region, field )
```