The four J ODEs

January-24-13
7:27 AM

why can't I do something easier?

Posterior J, h condition:

$$J_u(bch(\lambda_x,\lambda_y)) = J_u(\lambda_x)//CC_u^{\lambda_y//CC_u^{\lambda_x}} + J_u(\lambda_y//CC_u^{\lambda_x})$$

Using $\lambda_x = s\lambda$, $\lambda_y = \epsilon\lambda$, infinitesimal ϵ,
$\lambda_s := \lambda//CC_u^{s\lambda}$, get

$$\frac{d}{ds}J(s) = J(s)//der(u\to[\lambda_s,u]) + div_u\,\lambda_s$$

Posterior J, t condition:

$$J_w(\lambda/u,v\to w) = \left[J_u(\lambda)//CC_v^{\lambda//CC_u^{\lambda}} + J_v(\lambda//CC_u^{\lambda})\right]//u,v\to w$$

Using $\lambda = \lambda//w\to su+\epsilon v$, infinitesimal ϵ ?

Prior J [meaning P], h condition:

$$P_u(bch(\lambda_x,\lambda_y))//CC_u^{\lambda_x} = P_u(\lambda_x)//CC_u^{\lambda_x} + P_u(\lambda_y//CC_u^{\lambda_x})$$

Using $\lambda_x = s\lambda$, $\lambda_y = \epsilon\lambda$, infinitesimal ϵ, get

$$\left(\frac{d}{ds}P(s)\right)//CC_u^{s\lambda} = div_u(\lambda//CC_u^{s\lambda})$$

Test that!

Aside: what is $(CC_u^{\lambda})^{-1}$? Isn't it a simple-minded conjugation?

Some A-T Notions. $\mathfrak{a}_n$ is the vector space with basis $x_1,\ldots,x_n$, $\mathfrak{lie}_n = \mathfrak{lie}(\mathfrak{a}_n)$ is the free Lie algebra, $\mathrm{Ass}_n = \mathcal{U}(\mathfrak{lie}_n)$ is the free associative algera "of words", $\mathrm{tr}: \mathrm{Ass}_n^+ \to \mathrm{tr}_n = \mathrm{Ass}_n^+/(x_{i_1}x_{i_2}\cdots x_{i_m} = x_{i_2}\cdots x_{i_m}x_{i_1})$ is the "trace" into "cyclic words", $\mathfrak{der}_n = \mathfrak{der}(\mathfrak{lie}_n)$ are all the derivations, and

Always good to compare - from

$\mathcal{U}(\mathfrak{lie}_n)$ is the free associative algera "of words", $\mathrm{tr} : \mathrm{Ass}_n^+ \to \mathfrak{tr}_n = \mathrm{Ass}_n^+/(x_{i_1}x_{i_2}\cdots x_{i_m} = x_{i_2}\cdots x_{i_m}x_{i_1})$ is the "trace" into "cyclic words", $\mathfrak{der}_n = \mathfrak{der}(\mathfrak{lie}_n)$ are all the derivations, and

$$\mathfrak{tder}_n = \{D \in \mathfrak{der}_n : \forall i\ \exists a_i \text{ s.t. } D(x_i) = [x_i, a_i]\}$$

are "tangential derivations", so $D \leftrightarrow (a_1, \ldots, a_n)$ is a vector space isomorphism $\mathfrak{a}_n \oplus \mathfrak{tder}_n \cong \bigoplus_n \mathfrak{lie}_n$. Finally, $\mathrm{div} : \mathfrak{tder}_n \to \mathfrak{tr}_n$ is $(a_1, \ldots, a_n) \mapsto \sum_k \mathrm{tr}(x_k(\partial_k a_k))$, where for $a \in \mathrm{Ass}_n^+$, $\partial_k a \in \mathrm{Ass}_n$ is determined by $a = \sum_k (\partial_k a) x_k$, and $j : \mathrm{TAut}_n = \exp(\mathfrak{tder}_n) \to \mathfrak{tr}_n$ is $j(e^D) = \frac{e^D-1}{D} \cdot \mathrm{div}\, D$.

Theorem. Everything matches. $\langle$trees$\rangle$ is $\mathfrak{a}_n \oplus \mathfrak{tder}_n$ as Lie algebras, $\langle$wheels$\rangle$ is $\mathfrak{tr}_n$ as $\langle$trees$\rangle$ / $\mathfrak{tder}_n$-modules, $\mathrm{div}\, D = \iota^{-1}(u-l)(D)$, and $e^{uD}e^{-lD} = e^{jD}$.

Differential Operators. Interpret $\hat{\mathcal{U}}(I\mathfrak{g})$ as tangential differential operators on $\mathrm{Fun}(\mathfrak{g})$:

- $\varphi \in \mathfrak{g}^*$ becomes a multiplication operator.
- $x \in \mathfrak{g}$ becomes a tangential derivation, in the direction of the action of $\mathrm{ad}\, x$: $(x\varphi)(y) := \varphi([x, y])$.

Trees become vector fields and $uD \mapsto lD$ is $D \mapsto D^*$. So $\mathrm{div}\, D$ is $D - D^*$ and $jD = \log(e^D(e^D)^*) = \int_0^1 dt e^{tD}\, \mathrm{div}\, D$.

Compare — from the Montpelier handout.

Prior J [meaning P], t condition:

$$tm^{uv}_w // tha^{wx} = tha^{ux} // tha^{vx} // tm^{uv}_w$$

Given λ_x, this becomes

$$P_w(\lambda_x // u,v\to w) // RC_w^{\lambda_x // u,v\to w} =$$

$$P_u(\lambda_x) // RC_u^{\lambda_x} // RC_v^{\lambda_x // RC_u^{\lambda_x}} // (u,v\to w)$$

$$+ P_v(\lambda_x // RC_u^{\lambda_x}) // RC_v^{\lambda_x // RC_u^{\lambda_x}} // (u,v\to w)$$

Remember,

$$\lambda // (u,v\to w) // RC_w^{\lambda_x // (u,v\to w)} = \lambda // RC_u^{\lambda_x} // RC_v^{\lambda_x // RC_u^{\lambda_x}} // (u,v\to w)$$

or

$$(u,v\to w) // C_w^{-\lambda_x // (u,v\to w)} = C_u^{-\lambda_x // RC_v^{\lambda_x}} // C_v^{-\lambda_x} // (u,v\to w)$$

I wish I really understood the rules of the game...

Aside: $(u,v\to w) // \mathrm{div}_w = \mathrm{div}_u // (u,v\to w)$

$+ \; div_v \; // \; (u, v \to w)$