

## Evolving the Incremental $\lambda$ Calculus into a Model of Forward AD

Barak A. Pearlmutter<sup>1</sup> Robert Kelly<sup>1</sup> <sup>1</sup>Dept of Computer Science, Maynooth University <sup>2</sup>School of Electrical and Computer Engineering, Purdue University

## Key Idea

- Formal transformations resembling derivatives common in CS Derivatives of regular expressions (Brzozowski, 1964)
- Derivatives of types (McBride, 2001; Abbott et al., 2004)
- Incremental  $\lambda$ -Calculus (ILC; Cai et al., 2014)

Maynooth

University

National University

of Ireland Maynooth

## **Incremental** $\lambda$ -Calculus

- $\triangleright \lambda$  Calculus formalises function definition and application
- ILC adds  $\mathcal{D}$  to model incremental computation
- $\blacktriangleright \mathcal{D}$  maps a function  $f: B \rightarrow B$ 
  - which alters a database B
  - to an update function  $\mathcal{D}f: B \to \Delta B \to \Delta B$ 
    - where  $\Delta B$  is the type of *changes* to B
- Mechanically verifiable proofs of various properties

**ILC has Properties Resembling Calculus** 

 $\mathcal{D}(\lambda x \cdot f(g x)) = (\lambda x x' \cdot \mathcal{D} f(g x) (\mathcal{D} g x x'))$  $\mathcal{D}(f \circ g) x = \mathcal{D}f(g x) \circ \mathcal{D}g x$ 

## **Differences Between ILC and Forward AD**

- Propagates changes rather than tangents
- Changes are elements of *change sets*
- Changes are finite, not infinitesimal
- ▶ "Differences" ( $\Delta$ ), not "differentials" ( $\partial$ )
- Changes are separate arguments, tangents are bundled with primals
- ► ILC wishes to partial evaluate the primal away
- Forward AD does not

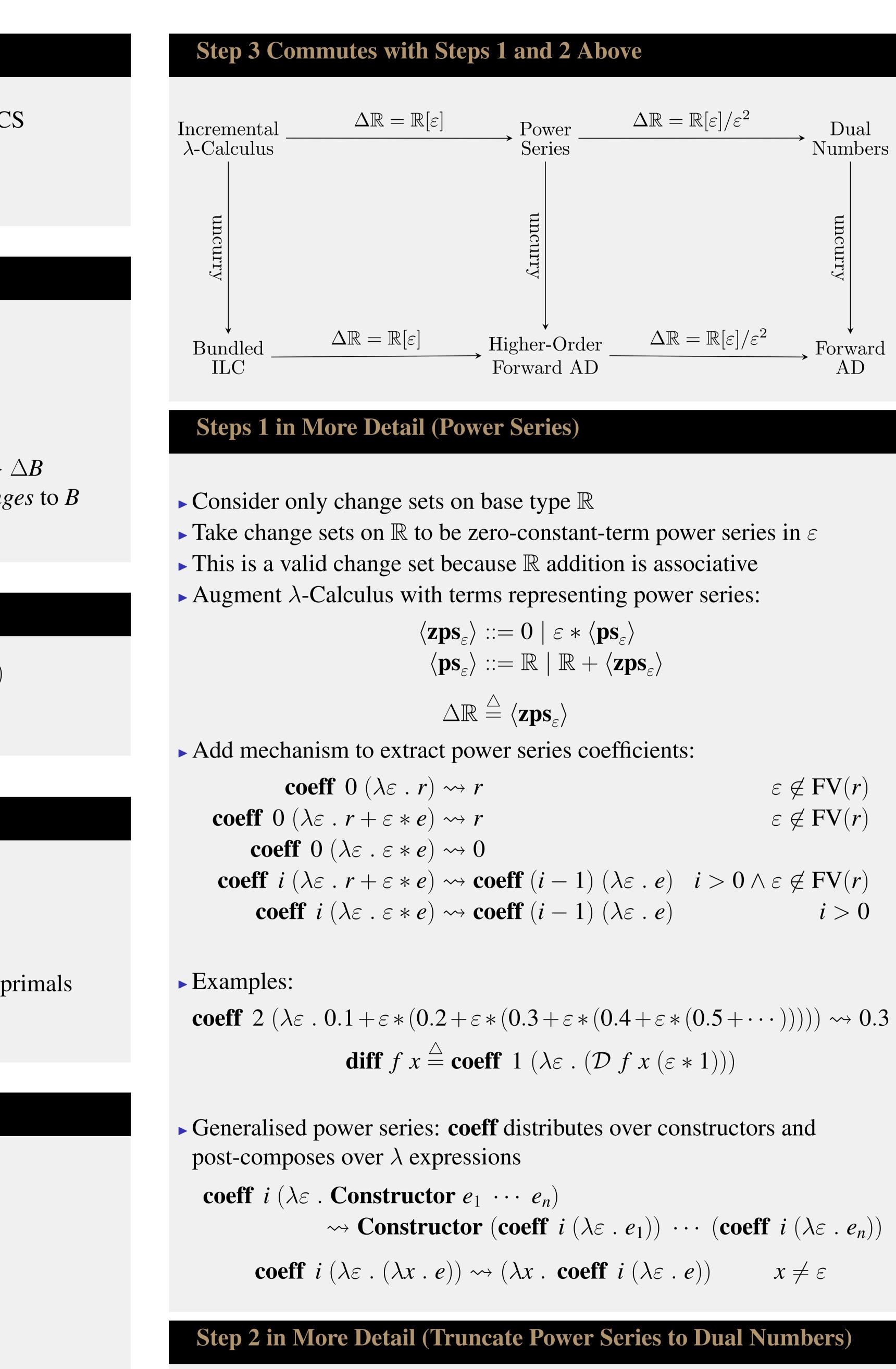
## **Reducing ILC to Forward AD in Three Steps**

- . Take the change sets of  $\mathbb{R}$  to be *power series* over  $\mathbb{R}$
- 2. Truncate these power series to dual numbers
- 3. Uncurry and bundle

$$f(x)(x') \leadsto f(x, x') \leadsto f(\langle x, x' \rangle)$$

and also return primal

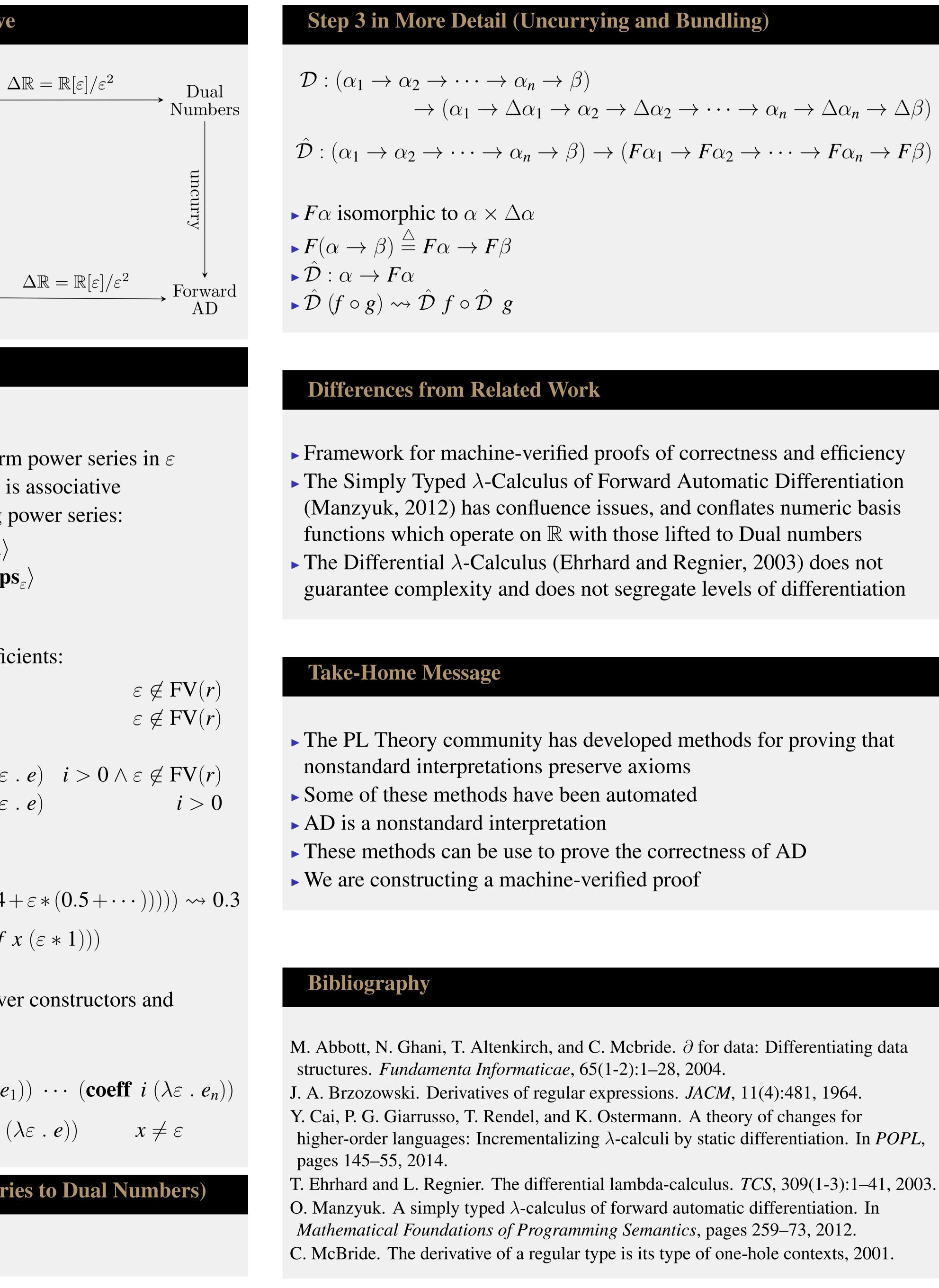
$$\mathcal{D}: (\alpha \to \beta) \to (\alpha \to \Delta \alpha \to \Delta \beta)$$
$$\rightsquigarrow \hat{\mathcal{D}}: (\alpha \to \beta) \to ((\alpha \times \Delta \alpha))$$



 $\rightarrow (\beta \times \Delta \beta))$ 

laziness

# Jeffrey Mark Siskind<sup>2</sup>







This work was supported, in part, by Science Foundation Ireland grant 09/IN.1/12637 and by NSF grant 1522954-IIS. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the