# Tactics for Natural Deduction in Coq

The Coq proof assistant is based on (predicative) Calculus of Inductive Constructions (CIC), a combination of a higher-order logic and a richly-typed functional programming language. CIC is very expressive and can encode the whole of (intuitionistic) first-order logic. Proof construction using Coq can also be carried out in a manner very much like that using natural deduction in the sequent form.

| | $\wedge$<br>`/\` | $\vee$<br>`\/` | $\rightarrow$<br>`->` | $\top$<br>*True* | $\bot$<br>*False* | $\neg$<br>`~` |
|---|---|---|---|---|---|---|
| Introduction | `split` | `left`<br>`right` | `intro`<br>`intro` $H$ | `exact I` | | `intro`<br>`intro` $H$ |
| Elimination | `apply` $H$<br>`elim` $H$ | `case` $H$<br>`elim` $H$ | `apply` $H$ | | `elim` $H$ | `apply` $H$<br>`elim` $H$ |

| | $\forall$<br>`forall` | $\exists$<br>`exists` | $=$<br>`=` |
|---|---|---|---|
| Introduction | `intro`<br>`intro` $H$ | `exists` $v$ | `reflexivity` |
| Elimination | `apply` $H$ | `apply` $H$<br>`elim` $H$ | `rewrite <-` $H$<br>`rewrite` $H$ |

(other tactics: `assumption`, `cut`, and `assert`, explained below)

In the above, $H$ names, or gives name to, an assumption (which is a "higher-order term") in the environment (i.e., the antecedent of the current sequent/goal); same for $v$ (in `exists` $v$). In Coq, $\neg A$ is written as `~A`, which is defined to be `A -> False` (i.e., $A \rightarrow \bot$).

Always end a tactic with a period "." so that Coq knows the proof command is completed. Say "`assumption`" when you find the current goal to be an axiom as in the rules of Natural Deduction (appended below). It may occur that you want to apply the $\rightarrow$-Elimination rule (see the appendix), while $A \rightarrow B$ is not immediately available in the set $\Gamma$ of assumptions. Use "`cut` $A$" in this case; enclose $A$ in parentheses if it is a compound formula. A similar tactic is "`assert` $A$," which you may find more convenient.

## Appendix

### Intuitionistic Natural Deduction

Below are the inference rules in the sequent form for intuitionistic first-order logic with equality.

$$\frac{}{A_1, \cdots, A_i, \cdots, A_n \vdash A_i}\ (Hyp^i)$$

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ (\wedge I) \qquad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}\ (\wedge E_1) \qquad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}\ (\wedge E_2)$$

$$\frac{\Gamma \vdash A}{\Gamma \vdash A \vee B}\ (\vee I_1) \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B}\ (\vee I_2) \qquad \frac{\Gamma \vdash A \vee B \qquad \Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma \vdash C}\ (\vee E)$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}\ (\to I) \qquad \frac{\Gamma \vdash A \to B \qquad \Gamma \vdash A}{\Gamma \vdash B}\ (\to E)$$

$$\frac{}{\Gamma \vdash \top}\ (\top I)$$

$$\frac{\Gamma \vdash \bot}{\Gamma \vdash A}\ (\bot E)$$

Note: the last rule says that, if we can deduce $\bot$ (or *False* , representing a contradiction), then we can deduce anything.

Note: with the $\to$-Elimination rule, one can deduce a contradiction (and hence anything) from $\neg A$ (or $A \to \bot$) and $A$.

$$\frac{\Gamma \vdash A[y/x]}{\Gamma \vdash \forall x A}\ (\forall I) \qquad \frac{\Gamma \vdash \forall x A}{\Gamma \vdash A[t/x]}\ (\forall E)$$

$$\frac{\Gamma \vdash A[t/x]}{\Gamma \vdash \exists x A}\ (\exists I) \qquad \frac{\Gamma \vdash \exists x A \qquad \Gamma, A[y/x] \vdash B}{\Gamma \vdash B}\ (\exists E)$$

Note: in the quantifier rules above, we assume that all substitutions are admissible and $y$ does not occur free in $\Gamma$ or $A$.

Let $t, t_1, t_2$ be arbitrary terms and again assume all substitutions are admissible.

$$\frac{}{\Gamma \vdash t = t}\ (= I) \qquad \frac{\Gamma \vdash t_1 = t_2 \qquad \Gamma \vdash A[t_1/x]}{\Gamma \vdash A[t_2/x]}\ (= E)$$

Note: the $=$ sign is part of the object language, not a meta symbol.