*(In the examples, the temperature is measured before the set of the speed).*

**Completely defined by the user:**

Just follow the points defined by the user (using interpolation).

**Maximum power-efficiency (big loop time)**

The user defines the target temperature; the controller starts from e.g. 50% and it adjust the speed wrt the measured temperature. Linear delta rules to maximize prediction. If the temperature is under the target, but it’s growing we must put a rule to say that every certain number of growing step we should put a little growing of speed e.g. 1%. Growing steps over the target have e.g. 2% of increment. Here we don’t need to memorize the. When it’s on the target with growing behavior we are going to put a little step.

(e.g. target 75)

50% 70

50% 71

52% 72

52% 73

53% 74

54% 75

56% 77

58% 78

…

62% 76

65% 75

65% 75

65% 74

65% 72

64% 71

64% 70

63% 68

64% 69

…

62% 68

62% 68

63% 68

63% 69

63% 69

64% 70

…

**Maximum cooling-efficiency (little loop time)**

The controller starts at 100% and when the temperature is stable (i.e. several identical results -> bht) or is descendent under the target one, it slowly go down until the maximum stable temperature that is under the target. Lowering steps are little (e.g. 1%) when growing steps are big (e.g. 10% if over the critical, 5% if under). Also need to put non-linear rules on the steps to take, depending on the delta of temperature.

Need to avoid loops by memorizing the behavior near the critical point:

(e.g. target of 25)

100% 50°

100% 50°

100% 49°

100% 47°

…

100% 25

100% 24 (not yet because we neither know if it’s stable nor lowering)

99% 23

…

91% 20

96% 21

96% 20

95% 19

…

**Industrial setting for equal-daily behavior**

Can exploit a time-based behavior (also automatic).

(e.g. first day everything 100%, second day -5% in unchanged areas, third day -5% in unchanged areas, when there aren’t unchanged areas it lowers the overall offset and so on – lowering the “roof” automatically).

**Industrial setting for equal-sequence-behavior**

BHT