General Integration Rules

Piecewise Constant Extraction Rules

- **Derivation: Piecewise constant extraction**
- **Basis:** $\partial_z \frac{(a\, f[z]^m)^q}{f[z]^{m\,q}} = 0$
- **Note: It is better to use trig substitution for integrands of the form** $\left(f[c+d]^2\right)^p$ **where** f **is a secant or cosecant and** p **is a positive half integer.**
- **Rule: If** $m\,p \in \mathbb{Z} \wedge p = r + q \wedge r \in \mathbb{Z}$**, then**

$$\int u\,(a\,v^m)^p\,dx \longrightarrow a^r\,\frac{(a\,v^m)^q}{v^{m\,q}}\int u\,v^{m\,p}\,dx$$

- **Program code:**

```
Int[u_.*(a_.*v_^m_)^p_, x_Symbol] :=
  Module[{q=FractionalPart[p]},
  a^(p-q)*(a*v^m)^q/v^(m*q)*Int[u*v^(m*p),x]] /;
FreeQ[{a,m,p},x] && IntegerQ[m*p] &&
  Not[MatchQ[u*(a*v^m)^p,(Sec[c_.+d_.*x]^2)^p /; FreeQ[{c,d},x] && p>0]] &&
  Not[MatchQ[u*(a*v^m)^p,(Csc[c_.+d_.*x]^2)^p /; FreeQ[{c,d},x] && p>0]] &&
  Not[MatchQ[u*(a*v^m)^p,(Sech[c_.+d_.*x]^2)^p /; FreeQ[{c,d},x] && p>0]] &&
  Not[MatchQ[u*(a*v^m)^p,(-Csch[c_.+d_.*x]^2)^p /; FreeQ[{c,d},x] && p>0]]
```

- **Derivation: Power rule for integration**
- **Basis:** $\partial_x \frac{(a\, f[x]^m\, g[x]^n)^p}{f[x]^{m\,p}\, g[x]^{n\,p}} = 0$
- **Rule: If** $m\,p \in \mathbb{Z} \wedge n\,p \in \mathbb{Z} \wedge p = r + q \wedge r \in \mathbb{Z}$**, then**

$$\int u\,(a\,v^m\,w^n)^p\,dx \longrightarrow a^r\,\frac{(a\,v^m\,w^n)^q}{v^{m\,q}\,w^{n\,q}}\int u\,v^{m\,p}\,w^{n\,p}\,dx$$

- **Program code:**

```
Int[u_.*(a_.*v_^m_.*w_^n_.)^p_, x_Symbol] :=
  Module[{q=FractionalPart[p]},
  a^(p-q)*(a*v^m*w^n)^q/(v^(m*q)*w^(n*q))*Int[u*v^(m*p)*w^(n*p),x]] /;
FreeQ[{a,m,n,p},x] && IntegerQ[m*p] && IntegerQ[n*p]
```

Derivative Divides Rules

- Derivation: Integration by substitution
- Basis: Int[f[g[x]]*g'[x], x] == Subst[Int[f[x], x], x, g[x]]

```
Int[u_*x_^m_.,x_Symbol] :=
  Dist[1/(m+1),Subst[Int[Regularize[SubstFor[x^(m+1),u,x],x],x],x,x^(m+1)]] /;
FreeQ[m,x] && NonzeroQ[m+1] && FunctionOfQ[x^(m+1),u,x]
```

- Derivation: Integration by substitution

```
Int[u_*Log[v_],x_Symbol] :=
  Module[{w=DerivativeDivides[v,u*(1-v),x]},
  w*PolyLog[2,1-v] /;
 Not[FalseQ[w]]]
```

- Derivation: Integration by substitution

```
Int[u_*PolyLog[n_,v_],x_Symbol] :=
  Module[{w=DerivativeDivides[v,u*v,x]},
  w*PolyLog[n+1,v] /;
 Not[FalseQ[w]]] /;
FreeQ[n,x]
```

- Derivation: Integration by substitution
- Basis: Int[f[g[x]]*g'[x], x] == Subst[Int[f[x], x], x, g[x]]

```
If[ShowSteps,

Int[u_*f_[a1___,g_[b1___,h_[c1___,v_,c2___],b2___],a2___],x_Symbol] :=
  Module[{z=DerivativeDivides[v,u,x]},
  ShowStep["","Int[f[g[x]]*g'[x],x]","Subst[Int[f[x],x],x,g[x]]",Hold[
  Dist[z,Subst[Int[f[a1,g[b1,h[c1,x,c2],b2],a2],x],x,v]]]] /;
 Not[FalseQ[z]]] /;
SimplifyFlag && FreeQ[{a1,a2,b1,b2,c1,c2,f,g},x],

Int[u_*f_[a1___,g_[b1___,h_[c1___,v_,c2___],b2___],a2___],x_Symbol] :=
  Module[{z=DerivativeDivides[v,u,x]},
  Dist[z,Subst[Int[f[a1,g[b1,h[c1,x,c2],b2],a2],x],x,v]] /;
 Not[FalseQ[z]]] /;
SimplifyFlag && FreeQ[{a1,a2,b1,b2,c1,c2,f,g},x]]
```

- Derivation: Integration by substitution
- Basis: Int[f[g[x]]*g'[x], x] == Subst[Int[f[x], x], x, g[x]]

```
If[ShowSteps,

Int[u_*f_[a1___,g_[b1___,v_,b2___],a2___],x_Symbol] :=
  Module[{z=DerivativeDivides[v,u,x]},
  ShowStep["","Int[f[g[x]]*g'[x],x]","Subst[Int[f[x],x],x,g[x]]",Hold[
  Dist[z,Subst[Int[f[a1,g[b1,x,b2],a2],x],x,v]]]] /;
 Not[FalseQ[z]]] /;
SimplifyFlag && FreeQ[{a1,a2,b1,b2,f,g},x],

Int[u_*f_[a1___,g_[b1___,v_,b2___],a2___],x_Symbol] :=
  Module[{z=DerivativeDivides[v,u,x]},
  Dist[z,Subst[Int[f[a1,g[b1,x,b2],a2],x],x,v]] /;
 Not[FalseQ[z]]] /;
SimplifyFlag && FreeQ[{a1,a2,b1,b2,f,g},x]]
```

- Derivation: Integration by substitution
- Basis: Int[f[g[x]]*g'[x], x] == Subst[Int[f[x], x], x, g[x]]

```
If[ShowSteps,

Int[u_*f_[a1___,v_,a2___],x_Symbol] :=
  Module[{z=DerivativeDivides[v,u,x]},
  ShowStep["","Int[f[g[x]]*g'[x],x]","Subst[Int[f[x],x],x,g[x]]",Hold[
  Dist[z,Subst[Int[f[a1,x,a2],x],x,v]]]] /;
 Not[FalseQ[z]]] /;
SimplifyFlag && FreeQ[{a1,a2,f},x],

Int[u_*f_[a1___,v_,a2___],x_Symbol] :=
  Module[{z=DerivativeDivides[v,u,x]},
  Dist[z,Subst[Int[f[a1,x,a2],x],x,v]] /;
 Not[FalseQ[z]]] /;
SimplifyFlag && FreeQ[{a1,a2,f},x]]
```

- Derivation: Integration by substitution
- Basis: Int[g[x]*g'[x], x] == Subst[Int[x, x], x, g[x]]

```
If[ShowSteps,

Int[u_*v_,x_Symbol] :=
  Module[{z=DerivativeDivides[v,u,x]},
  ShowStep["","Int[g[x]*g'[x],x]","Subst[Int[x,x],x,g[x]]",Hold[
  Dist[z,Subst[Int[x,x],x,v]]]] /;
 Not[FalseQ[z]]] /;
SimplifyFlag,

Int[u_*v_,x_Symbol] :=
  Module[{z=DerivativeDivides[v,u,x]},
  Dist[z,Subst[Int[x,x],x,v]] /;
 Not[FalseQ[z]]]]
```

- Derivation: Integration by substitution
- Basis: If n!=-1, Int[f[x]^n*g[x]^n*D[f[x]*g[x], x], x] == f[x]^(n+1)*g[x]^(n+1)/(n+1)
- Note: Need to generalize for any number of u' s raised to multiples of n!

```
If[ShowSteps,

Int[u1_^n_*u2_^n_*v_,x_Symbol] :=
  Module[{w=DerivativeDivides[u1*u2,v,x]},
  ShowStep["If nonzero[n+1],","Int[f[x]^n*g[x]^n*D[f[x]*g[x],x],x]",
                                 "f[x]^(n+1)*g[x]^(n+1)/(n+1)",Hold[
  w*u1^(n+1)*u2^(n+1)/(n+1)]] /;
 Not[FalseQ[w]]] /;
SimplifyFlag && FreeQ[n,x] && NonzeroQ[n+1] && (SumQ[v] || NonsumQ[u1*u2] || NonzeroQ[n-1]),

Int[u1_^n_*u2_^n_*v_,x_Symbol] :=
  Module[{w=DerivativeDivides[u1*u2,v,x]},
  w*u1^(n+1)*u2^(n+1)/(n+1) /;
 Not[FalseQ[w]]] /;
FreeQ[n,x] && NonzeroQ[n+1] && (SumQ[v] || NonsumQ[u1*u2] || NonzeroQ[n-1])]
```

- Derivation: Integration by substitution
- Basis: If n!=-1, Int[f[x]^n*g[x]^n*D[f[x]*g[x], x], x] == f[x]^(n+1)*g[x]^(n+1)/(n+1)

```
If[ShowSteps,

Int[x_^m_.*u_^n_.*v_,x_Symbol] :=
  Module[{w=DerivativeDivides[x*u,x^(m-n)*v,x]},
  ShowStep["If nonzero[n+1],","Int[f[x]^n*g[x]^n*D[f[x]*g[x],x],x]",
                                 "f[x]^(n+1)*g[x]^(n+1)/(n+1)",Hold[
  w*x^(n+1)*u^(n+1)/(n+1)]] /;
 Not[FalseQ[w]]] /;
SimplifyFlag && FreeQ[n,x] && NonzeroQ[n+1] && (SumQ[v] || NonsumQ[u] || NonzeroQ[n-1]),

Int[x_^m_.*u_^n_.*v_,x_Symbol] :=
  Module[{w=DerivativeDivides[x*u,x^(m-n)*v,x]},
  w*x^(n+1)*u^(n+1)/(n+1) /;
 Not[FalseQ[w]]] /;
FreeQ[n,x] && NonzeroQ[n+1] && (SumQ[v] || NonsumQ[u] || NonzeroQ[n-1])]
```

- Derivation: Integration by parts & power rule for integration
- Basis: If n!=-1, Int[x^m*f[x]^n*f'[x], x] == x^m*f[x]^(n+1)/(n+1) - m/(n+1)*Int[x^(m-1)*f[x]^(n+1)

```
If[ShowSteps,

Int[x_^m_.*u_^n_.*v_,x_Symbol] :=
  Module[{w=DerivativeDivides[u,v,x]},
  ShowStep["If nonzero[n+1],","Int[x^m*f[x]^n*f'[x],x]",
                 "x^m*f[x]^(n+1)/(n+1) - m/(n+1)*Int[x^(m-1)*f[x]^(n+1),x]",Hold[
  w*x^m*u^(n+1)/(n+1) -
    Dist[m/(n+1)*w,Int[x^(m-1)*u^(n+1),x]]]] /;
 Not[FalseQ[w]]] /;
SimplifyFlag && FreeQ[n,x] && NonzeroQ[n+1] && IntegerQ[m] && m>0 &&
    (SumQ[v] || NonsumQ[u] || NonzeroQ[n-1]),

Int[x_^m_.*u_^n_.*v_,x_Symbol] :=
  Module[{w=DerivativeDivides[u,v,x]},
  w*x^m*u^(n+1)/(n+1) -
    Dist[m/(n+1)*w,Int[x^(m-1)*u^(n+1),x]] /;
 Not[FalseQ[w]]] /;
FreeQ[n,x] && NonzeroQ[n+1] && IntegerQ[m] && m>0 && (SumQ[v] || NonsumQ[u] || NonzeroQ[n-1])]
```

- Derivation: Integration by substitution
- Basis: Int[f[Int[g[x], x]]*g[x], x] == Subst[Int[f[x], x], x, Int[g[x]]]

```
Int[u_*v_,x_Symbol] :=
  Module[{w=Block[{ShowSteps=False,StepCounter=Null}, Int[v,x]]},
  Subst[Int[Regularize[SubstFor[w,u,x],x],x],x,w] /;
 FunctionOfQ[w,u,x]] /;
SumQ[v] && PolynomialQ[v,x]
```

- Derivation: Integration by substitution
- Basis: Int[f[g[x]]*g'[x], x] == Subst[Int[f[x], x], x, g[x]]

```
Int[u_*(a_.+b_.*x_)^m_.,x_Symbol] :=
  Dist[1/(b*(m+1)),Subst[Int[Regularize[SubstFor[(a+b*x)^(m+1),u,x],x],x],x,(a+b*x)^(m+1)]] /;
FreeQ[{a,b,m},x] && NonzeroQ[m+1] && FunctionOfQ[(a+b*x)^(m+1),u,x] (* && NonsumQ[u] *)
```

- Derivation: Integration by substitution
- Basis: f[(c*x)^n]/x == f[(c*x)^n]/(n*(c*x)^n)*D[(c*x)^n,x]
- Basis: Int[f[(c*x)^n]/x, x] == Subst[Int[f[x]/x, x], x, (c*x)^n]/n

```
If[ShowSteps,

Int[u_/x_,x_Symbol] :=
  Module[{lst=PowerVariableExpn[u,0,x]},
  ShowStep["","Int[f[(c*x)^n]/x,x]","Subst[Int[f[x]/x,x],x,(c*x)^n]/n",Hold[
  Dist[1/lst[[2]],Subst[Int[Regularize[lst[[1]]/x,x],x],x,(lst[[3]]*x)^lst[[2]]]]]] /;
 Not[FalseQ[lst]] && NonzeroQ[lst[[2]]]] /;
SimplifyFlag && NonsumQ[u] && Not[RationalFunctionQ[u,x]],

Int[u_/x_,x_Symbol] :=
  Module[{lst=PowerVariableExpn[u,0,x]},
  Dist[1/lst[[2]],Subst[Int[Regularize[lst[[1]]/x,x],x],x,(lst[[3]]*x)^lst[[2]]]] /;
 Not[FalseQ[lst]] && NonzeroQ[lst[[2]]]] /;
NonsumQ[u] && Not[RationalFunctionQ[u,x]]]
```

- Derivation: Integration by substitution
- Basis: x^(n-1)*f[(c*x)^n] == f[(c*x)^n]/(c*n)*D[(c*x)^n,x]
- Basis: If g = GCD[m+1, n] > 1, Int[x^m*f[x^n], x] == Subst[Int[x^((m+1)/g-1)*f[x^(n/g)], x], x, x^g]/g

```
If[ShowSteps,

Int[u_*x_^m_.,x_Symbol] :=
  Module[{lst=PowerVariableExpn[u,m+1,x]},
  ShowStep["If g=GCD[m+1,n]>1,","Int[x^m*f[x^n],x]",
        "Subst[Int[x^((m+1)/g-1)*f[x^(n/g)],x],x,x^g]/g",Hold[
  Dist[1/lst[[2]],Subst[Int[Regularize[lst[[1]]/x,x],x],x,(lst[[3]]*x)^lst[[2]]]]]] /;
 NotFalseQ[lst] && NonzeroQ[lst[[2]]-m-1]] /;
SimplifyFlag && IntegerQ[m] && m!=-1 && NonsumQ[u] && (m>0 || Not[AlgebraicFunctionQ[u,x]]),

Int[u_*x_^m_.,x_Symbol] :=
  Module[{lst=PowerVariableExpn[u,m+1,x]},
  Dist[1/lst[[2]],Subst[Int[Regularize[lst[[1]]/x,x],x],x,(lst[[3]]*x)^lst[[2]]]] /;
 NotFalseQ[lst] && NonzeroQ[lst[[2]]-m-1]] /;
IntegerQ[m] && m!=-1 && NonsumQ[u] && (m>0 || Not[AlgebraicFunctionQ[u,x]])]
```

Trig Product Expansion Rules

- Derivation: Algebraic expansion

```
Int[u_,x_Symbol] :=
  Int[NormalForm[Expand[TrigReduce[u],x],x],x] /;
ProductQ[u] && Catch[Scan[Function[If[Not[LinearSinCosQ[#,x]],Throw[False]]],u];True]
```

```
LinearSinCosQ[u_^n_.,x_Symbol] :=
  IntegerQ[n] && n>0 && (SinQ[u] || CosQ[u]) && LinearQ[u[[1]],x]
```

Hyperbolic Product Expansion Rules

- Derivation: Algebraic expansion

```
Int[u_,x_Symbol] :=
  Int[NormalForm[Expand[TrigReduce[u],x],x],x] /;
ProductQ[u] && Catch[Scan[Function[If[Not[LinearSinhCoshQ[#,x]],Throw[False]]],u];True]
```

```
LinearSinhCoshQ[u_^n_.,x_Symbol] :=
  IntegerQ[n] && n>0 && (SinhQ[u] || CoshQ[u]) && LinearQ[u[[1]],x]
```

Impure Trig Substitution Rules

- Derivation: Integration by substitution
- Basis: f[Cos[z]]*Sin[z] == -f[Cos[z]] * Cos'[z]

```
Int[u_*Sin[c_.*(a_.+b_.*x_)],x_Symbol] :=
  -Dist[1/(b*c),Subst[Int[Regularize[SubstFor[Cos[c*(a+b*x)],u,x],x],x],x,Cos[c*(a+b*x)]]] /;
FreeQ[{a,b,c},x] && FunctionOfQ[Cos[c*(a+b*x)],u,x]
```

- Derivation: Integration by substitution
- Basis: f[Sin[z]]*Cos[z] == f[Sin[z]] * Sin'[z]

```
Int[u_*Cos[c_.*(a_.+b_.*x_)],x_Symbol] :=
  Dist[1/(b*c),Subst[Int[Regularize[SubstFor[Sin[c*(a+b*x)],u,x],x],x],x,Sin[c*(a+b*x)]]] /;
FreeQ[{a,b,c},x] && FunctionOfQ[Sin[c*(a+b*x)],u,x]
```

- Derivation: Integration by substitution
- Basis: f[Cos[z]]*Tan[z] == -f[Cos[z]]/Cos[z] * Cos'[z]

```
Int[u_*Tan[c_.*(a_.+b_.*x_)],x_Symbol] :=
  -Dist[1/(b*c),Subst[Int[Regularize[SubstFor[Cos[c*(a+b*x)],u,x]/x,x],x],x,Cos[c*(a+b*x)]]] /;
FreeQ[{a,b,c},x] && FunctionOfQ[Cos[c*(a+b*x)],u,x]
```

- Derivation: Integration by substitution
- Basis: f[Sin[z]]*Cot[z] == f[Sin[z]]/Sin[z] * Sin'[z]

```
Int[u_*Cot[c_.*(a_.+b_.*x_)],x_Symbol] :=
  Dist[1/(b*c),Subst[Int[Regularize[SubstFor[Sin[c*(a+b*x)],u,x]/x,x],x],x,Sin[c*(a+b*x)]]] /;
FreeQ[{a,b,c},x] && FunctionOfQ[Sin[c*(a+b*x)],u,x]
```

- Derivation: Integration by substitution
- Basis: If n is even, f[Tan[z]]*Sec[z]^n == f[Tan[z]]*(1+Tan[z]^2)^((n-2)/2) * Tan'[z]

```
Int[u_*Sec[c_.*(a_.+b_.*x_)]^n_,x_Symbol] :=
  Dist[1/(b*c),
    Subst[Int[Regularize[(1+x^2)^((n-2)/2)*SubstFor[Tan[c*(a+b*x)],u,x],x],x],x,Tan[c*(a+b*x)]]] /;
FreeQ[{a,b,c},x] && EvenQ[n] && FunctionOfQ[Tan[c*(a+b*x)],u,x] && NonsumQ[u]
```

- Derivation: Integration by substitution
- Basis: If n is even, f[Cot[z]]*Csc[z]^n == -f[Cot[z]]*(1+Cot[z]^2)^((n-2)/2) * Cot'[z]

```
Int[u_*Csc[c_.*(a_.+b_.*x_)]^n_,x_Symbol] :=
  -Dist[1/(b*c),
    Subst[Int[Regularize[(1+x^2)^((n-2)/2)*SubstFor[Cot[c*(a+b*x)],u,x],x],x],x,Cot[c*(a+b*x)]]] /;
FreeQ[{a,b,c},x] && EvenQ[n] && FunctionOfQ[Cot[c*(a+b*x)],u,x] && NonsumQ[u]
```

- Derivation: Integration by substitution
- Basis: f[Sin[z]]*Cos[z] == f[Sin[z]] * Sin'[z]

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{v=FunctionOfTrig[u,x]},
  ShowStep["","Int[f[Sin[a+b*x]]*Cos[a+b*x],x]","Subst[Int[f[x],x],x,Sin[a+b*x]]/b",Hold[
  Dist[1/Coefficient[v,x,1],Subst[Int[Regularize[SubstFor[Sin[v],u/Cos[v],x],x],x],x,Sin[v]]]]] /;
 NotFalseQ[v] && FunctionOfQ[Sin[v],u/Cos[v],x]] /;
SimplifyFlag,

Int[u_,x_Symbol] :=
  Module[{v=FunctionOfTrig[u,x]},
  Dist[1/Coefficient[v,x,1],Subst[Int[Regularize[SubstFor[Sin[v],u/Cos[v],x],x],x],x,Sin[v]]] /;
 NotFalseQ[v] && FunctionOfQ[Sin[v],u/Cos[v],x]]]
```

- Derivation: Integration by substitution
- Basis: f[Cos[z]]*Sin[z] == -f[Cos[z]] * Cos'[z]

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{v=FunctionOfTrig[u,x]},
  ShowStep["","Int[f[Cos[a+b*x]]*Sin[a+b*x],x]","-Subst[Int[f[x],x],x,Cos[a+b*x]]/b",Hold[
  -Dist[1/Coefficient[v,x,1],Subst[Int[Regularize[SubstFor[Cos[v],u/Sin[v],x],x],x],x,Cos[v]]]]] /;
 NotFalseQ[v] && FunctionOfQ[Cos[v],u/Sin[v],x]] /;
SimplifyFlag,

Int[u_,x_Symbol] :=
  Module[{v=FunctionOfTrig[u,x]},
  -Dist[1/Coefficient[v,x,1],Subst[Int[Regularize[SubstFor[Cos[v],u/Sin[v],x],x],x],x,Cos[v]]] /;
 NotFalseQ[v] && FunctionOfQ[Cos[v],u/Sin[v],x]]]
```

- Derivation: Integration by substitution
- Basis: f[Log[Tan[z]]]*Sec[z]*Csc[z] == f[Log[Tan[z]]] * D[Log[Tan[z]], z]

```
Int[u_*Sec[a_.+b_.*x_]*Csc[a_.+b_.*x_],x_Symbol] :=
  Dist[1/b,Subst[Int[Regularize[SubstFor[Log[Tan[a+b*x]],u,x],x],x],x,Log[Tan[a+b*x]]]] /;
FreeQ[{a,b},x] && FunctionOfQ[Log[Tan[a+b*x]],u,x]
```

- Derivation: Integration by substitution
- Basis: f[Log[Cot[z]]]*Sec[z]*Csc[z] == -f[Log[Cot[z]]] * D[Log[Cot[z]], z]

```
Int[u_*Sec[a_.+b_.*x_]*Csc[a_.+b_.*x_],x_Symbol] :=
  -Dist[1/b,Subst[Int[Regularize[SubstFor[Log[Cot[a+b*x]],u,x],x],x],x,Log[Cot[a+b*x]]]] /;
FreeQ[{a,b},x] && FunctionOfQ[Log[Cot[a+b*x]],u,x]
```

- Derivation: Integration by substitution
- Basis: f[Cos[z/2]*Sin[z/2]]*Cos[z] == 2*f[Cos[z/2]*Sin[z/2]] * D[Cos[z/2]*Sin[z/2], z]

```
Int[u_*Cos[a_.+b_.*x_],x_Symbol] :=
  Dist[2/b,Subst[Int[Regularize[SubstFor[Cos[a/2+b/2*x]*Sin[a/2+b/2*x],u,x],x],x],x,
                Cos[a/2+b/2*x]*Sin[a/2+b/2*x]]] /;
NonsumQ[u] && FreeQ[{a,b},x] && FunctionOfQ[Cos[a/2+b/2*x]*Sin[a/2+b/2*x],u,x]
```

Impure Hyperbolic Substitution Rules

- Derivation: Integration by substitution
- Basis: f[Cosh[z]]*Sinh[z] == f[Cosh[z]] * Cosh'[z]

```
Int[u_*Sinh[c_.*(a_.+b_.*x_)],x_Symbol] :=
  Dist[1/(b*c),Subst[Int[Regularize[SubstFor[Cosh[c*(a+b*x)],u,x],x],x],x,Cosh[c*(a+b*x)]]] /;
FreeQ[{a,b,c},x] && FunctionOfQ[Cosh[c*(a+b*x)],u,x]
```

- Derivation: Integration by substitution
- Basis: f[Sinh[z]]*Cosh[z] == f[Sinh[z]] * Sinh'[z]

```
Int[u_*Cosh[c_.*(a_.+b_.*x_)],x_Symbol] :=
  Dist[1/(b*c),Subst[Int[Regularize[SubstFor[Sinh[c*(a+b*x)],u,x],x],x],x,Sinh[c*(a+b*x)]]] /;
FreeQ[{a,b,c},x] && FunctionOfQ[Sinh[c*(a+b*x)],u,x]
```

- Derivation: Integration by substitution
- Basis: f[Cosh[z]]*Tanh[z] == f[Cosh[z]]/Cosh[z] * Cosh'[z]

```
Int[u_*Tanh[c_.*(a_.+b_.*x_)],x_Symbol] :=
  Dist[1/(b*c),Subst[Int[Regularize[SubstFor[Cosh[c*(a+b*x)],u,x]/x,x],x],x,Cosh[c*(a+b*x)]]] /;
FreeQ[{a,b,c},x] && FunctionOfQ[Cosh[c*(a+b*x)],u,x]
```

- Derivation: Integration by substitution
- Basis: f[Sinh[z]]*Coth[z] == f[Sinh[z]]/Sinh[z] * Sinh'[z]

```
Int[u_*Coth[c_.*(a_.+b_.*x_)],x_Symbol] :=
  Dist[1/(b*c),Subst[Int[Regularize[SubstFor[Sinh[c*(a+b*x)],u,x]/x,x],x],x,Sinh[c*(a+b*x)]]] /;
FreeQ[{a,b,c},x] && FunctionOfQ[Sinh[c*(a+b*x)],u,x]
```

- Derivation: Integration by substitution
- Basis: If n is even, f[Tanh[z]]*Sech[z]^n == f[Tanh[z]]*(1-Tanh[z]^2)^((n-2)/2) * Tanh'[z]

```
Int[u_*Sech[c_.*(a_.+b_.*x_)]^n_,x_Symbol] :=
  Dist[1/(b*c),
    Subst[Int[Regularize[(1-x^2)^((n-2)/2)*SubstFor[Tanh[c*(a+b*x)],u,x],x],x],x,Tanh[c*(a+b*x)]]] /
FreeQ[{a,b,c},x] && EvenQ[n] && FunctionOfQ[Tanh[c*(a+b*x)],u,x] && NonsumQ[u]
```

- Derivation: Integration by substitution
- Basis: If n is even, f[Coth[z]]*Csch[z]^n == -f[Coth[z]]*(-1+Coth[z]^2)^((n-2)/2) * Coth'[z]

```
Int[u_*Csch[c_.*(a_.+b_.*x_)]^n_,x_Symbol] :=
  -Dist[1/(b*c),
    Subst[Int[Regularize[(-1+x^2)^((n-2)/2)*SubstFor[Coth[c*(a+b*x)],u,x],x],x],x,Coth[c*(a+b*x)]]]
FreeQ[{a,b,c},x] && EvenQ[n] && FunctionOfQ[Coth[c*(a+b*x)],u,x] && NonsumQ[u]
```

- Derivation: Integration by substitution
- Basis: Int[f[Sinh[a+b*x]]*Cosh[a+b*x], x] == Subst[Int[f[x], x], x, Sinh[a+b*x]]/b

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{v=FunctionOfHyperbolic[u,x]},
  ShowStep["","Int[f[Sinh[a+b*x]]*Cosh[a+b*x],x]","Subst[Int[f[x],x],x,Sinh[a+b*x]]/b",Hold[
  Dist[1/Coefficient[v,x,1],Subst[Int[Regularize[SubstFor[Sinh[v],u/Cosh[v],x],x],x],x,Sinh[v]]]]] /;
 NotFalseQ[v] && FunctionOfQ[Sinh[v],u/Cosh[v],x]] /;
SimplifyFlag,

Int[u_,x_Symbol] :=
  Module[{v=FunctionOfHyperbolic[u,x]},
  Dist[1/Coefficient[v,x,1],Subst[Int[Regularize[SubstFor[Sinh[v],u/Cosh[v],x],x],x],x,Sinh[v]]] /;
 NotFalseQ[v] && FunctionOfQ[Sinh[v],u/Cosh[v],x]]]
```

- Derivation: Integration by substitution
- Basis: Int[f[Cosh[a+b*x]]*Sinh[a+b*x], x] == Subst[Int[f[x], x], x, Cosh[a+b*x]]/b

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{v=FunctionOfHyperbolic[u,x]},
  ShowStep["","Int[f[Cosh[a+b*x]]*Sinh[a+b*x],x]","Subst[Int[f[x],x],x,Cosh[a+b*x]]/b",Hold[
  Dist[1/Coefficient[v,x,1],Subst[Int[Regularize[SubstFor[Cosh[v],u/Sinh[v],x],x],x],x,Cosh[v]]]]] /;
 NotFalseQ[v] && FunctionOfQ[Cosh[v],u/Sinh[v],x] (* && Not[FunctionOfQ[Tanh[v],u,x]] *)] /;
SimplifyFlag,

Int[u_,x_Symbol] :=
  Module[{v=FunctionOfHyperbolic[u,x]},
  Dist[1/Coefficient[v,x,1],Subst[Int[Regularize[SubstFor[Cosh[v],u/Sinh[v],x],x],x],x,Cosh[v]]] /;
 NotFalseQ[v] && FunctionOfQ[Cosh[v],u/Sinh[v],x] (* && Not[FunctionOfQ[Tanh[v],u,x]] *)]]
```

- Derivation: Integration by substitution
- Basis: f[Log[Tanh[z]]]*Sech[z]*Csch[z] == f[Log[Tanh[z]]] * D[Log[Tanh[z]], z]

```
Int[u_*Sech[a_.+b_.*x_]*Csch[a_.+b_.*x_],x_Symbol] :=
  Dist[1/b,Subst[Int[Regularize[SubstFor[Log[Tanh[a+b*x]],u,x],x],x],x,Log[Tanh[a+b*x]]]] /;
FreeQ[{a,b},x] && FunctionOfQ[Log[Tanh[a+b*x]],u,x]
```

- Derivation: Integration by substitution
- Basis: f[Log[Coth[z]]]*Sech[z]*Csch[z] == -f[Log[Coth[z]]] * D[Log[Coth[z]], z]

```
Int[u_*Sech[a_.+b_.*x_]*Csch[a_.+b_.*x_],x_Symbol] :=
  -Dist[1/b,Subst[Int[Regularize[SubstFor[Log[Coth[a+b*x]],u,x],x],x],x,Log[Coth[a+b*x]]]] /;
FreeQ[{a,b},x] && FunctionOfQ[Log[Coth[a+b*x]],u,x]
```

- Derivation: Integration by substitution
- Basis: f[Cosh[z/2]*Sinh[z/2]]*Cosh[z] == 2*f[Cosh[z/2]*Sinh[z/2]] * D[Cosh[z/2]*Sinh[z/2], z]

```
Int[u_*Cosh[a_.+b_.*x_],x_Symbol] :=
  Dist[2/b,Subst[Int[Regularize[SubstFor[Cosh[a/2+b/2*x]*Sinh[a/2+b/2*x],u,x],x],x],x,
                 Cosh[a/2+b/2*x]*Sinh[a/2+b/2*x]]] /;
NonsumQ[u] && FreeQ[{a,b},x] && FunctionOfQ[Cosh[a/2+b/2*x]*Sinh[a/2+b/2*x],u,x]
```

Derivative Divides Rules

- Derivation: Integration by substitution
- Basis: Int[x^m*f[x]^(-1+a*x^m)*f'[x], x] == f[x]^(a*x^m)/a - m*Int[x^(m-1)*f[x]^(a*x^m)*Log[f[x]], x]

```
If[ShowSteps,

Int[x_^m_.*u_^(-1+a_.*x_^m_.)*v_,x_Symbol] :=
  Module[{w=DerivativeDivides[u,v,x]},
  ShowStep["If m>0,","Int[x^m*f[x]^(-1+a*x^m)*f'[x],x]",
                "f[x]^(a*x^m)/a - m*Int[x^(m-1)*f[x]^(a*x^m)*Log[f[x]],x]",Hold[
  w*u^(a*x^m)/a -
    Dist[m*w,Int[x^(m-1)*u^(a*x^m)*Log[u],x]]]] /;
 Not[FalseQ[w]]] /;
SimplifyFlag && FreeQ[a,x] && RationalQ[m] && m>0,

Int[x_^m_.*u_^(-1+a_.*x_^m_.)*v_,x_Symbol] :=
  Module[{w=DerivativeDivides[u,v,x]},
  w*u^(a*x^m)/a -
    Dist[m*w,Int[x^(m-1)*u^(a*x^m)*Log[u],x]] /;
 Not[FalseQ[w]]] /;
FreeQ[a,x] && RationalQ[m] && m>0]
```

Trig Substitution Rules

- Derivation: Integration by substitution
- Basis: f[Cot[z]] == -f[Cot[z]]/(1+Cot[z]^2) * Cot'[z]

```
(* If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{v=FunctionOfTrig[u,x]},
  ShowStep["","Int[f[Cot[a+b*x]],x]","-Subst[Int[f[x]/(1+x^2),x],x,Cot[a+b*x]]/b",Hold[
  -Dist[1/Coefficient[v,x,1],Subst[Int[Regularize[SubstFor[Cot[v],u,x]/(1+x^2),x],x],x,Cot[v]]]]] /;
 NotFalseQ[v] && FunctionOfQ[Cot[v],u,x]] /;
SimplifyFlag,

Int[u_,x_Symbol] :=
  Module[{v=FunctionOfTrig[u,x]},
  -Dist[1/Coefficient[v,x,1],Subst[Int[Regularize[SubstFor[Cot[v],u,x]/(1+x^2),x],x],x,Cot[v]]] /;
 NotFalseQ[v] && FunctionOfQ[Cot[v],u,x]]] *)
```

- Derivation: Integration by substitution
- Basis: f[Tan[z]] == f[Tan[z]]/(1+Tan[z]^2) * Tan'[z]

```
(* If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{v=FunctionOfTrig[u,x]},
  ShowStep["","Int[f[Tan[a+b*x]],x]","Subst[Int[f[x]/(1+x^2),x],x,Tan[a+b*x]]/b",Hold[
  Dist[1/Coefficient[v,x,1],Subst[Int[Regularize[SubstFor[Tan[v],u,x]/(1+x^2),x],x],x,Tan[v]]]]] /;
 NotFalseQ[v] && FunctionOfQ[Tan[v],u,x]] /;
SimplifyFlag,

Int[u_,x_Symbol] :=
  Module[{v=FunctionOfTrig[u,x]},
  Dist[1/Coefficient[v,x,1],Subst[Int[Regularize[SubstFor[Tan[v],u,x]/(1+x^2),x],x],x,Tan[v]]] /;
 NotFalseQ[v] && FunctionOfQ[Tan[v],u,x]]] *)
```

- Derivation: Integration by substitution
- Basis: f[Tan[z]] == f[Tan[z]]/(1+Tan[z]^2) * Tan'[z]

```
Int[u_,x_Symbol] :=
  Subst[Int[Regularize[SubstFor[Tan[x],u,x]/(1+x^2),x],x],x,Tan[x]] /;
FunctionOfQ[Tan[x],u,x] && FunctionOfTanWeight[u,x,x]>=0 && TryTanSubst[u,x]
```

- Derivation: Integration by substitution
- Basis: f[Cot[z]] == -f[Cot[z]]/(1+Cot[z]^2) * Cot'[z]

```
Int[u_,x_Symbol] :=
  -Subst[Int[Regularize[SubstFor[Cot[x],u,x]/(1+x^2),x],x],x,Cot[x]] /;
FunctionOfQ[Cot[x],u,x] && FunctionOfTanWeight[u,x,x]<0 && TryTanSubst[u,x]
```

Hyperbolic Substitution Rules

- Derivation: Integration by substitution
- Basis: f[Tanh[z]] == f[Tanh[z]] / (1-Tanh[z]^2) * Tanh'[z]

```
Int[u_,x_Symbol] :=
  Subst[Int[Regularize[SubstFor[Tanh[x],u,x]/(1-x^2),x],x],x,Tanh[x]] /;
FunctionOfQ[Tanh[x],u,x] && FunctionOfTanhWeight[u,x,x]>=0 && TryTanhSubst[u,x]
```

- Derivation: Integration by substitution
- Basis: f[Coth[z]] == f[Coth[z]] / (1-Coth[z]^2) * Coth'[z]

```
Int[u_,x_Symbol] :=
  Subst[Int[Regularize[SubstFor[Coth[x],u,x]/(1-x^2),x],x],x,Coth[x]] /;
FunctionOfQ[Coth[x],u,x] && FunctionOfTanhWeight[u,x,x]<0 && TryTanhSubst[u,x]
```

Exponential Substitution Rules

- Derivation: Integration by substitution
- Basis: Int[f[E^(a+b*x)], x] == Subst[Int[f[x]/x, x], x, E^(a+b*x)]/b
- Basis: Int[g[f^(a+b*x)], x] == Subst[Int[g[x]/x, x], x, f^(a+b*x)]/(b*Log[f])

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{lst=FunctionOfExponentialOfLinear[u,x]},
  If[lst[[4]]===E,
    ShowStep["","Int[f[E^(a+b*x)],x]","Subst[Int[f[x]/x,x],x,E^(a+b*x)]/b",Hold[
    Dist[1/lst[[3]],Subst[Int[Regularize[lst[[1]]/x,x],x],x,E^(lst[[2]]+lst[[3]]*x)]]]],
  ShowStep["","Int[g[f^(a+b*x)],x]","Subst[Int[g[x]/x,x],x,f^(a+b*x)]/(b*Log[f])",Hold[
  Dist[1/(lst[[3]]*Log[lst[[4]]]),
       Subst[Int[Regularize[lst[[1]]/x,x],x],x,lst[[4]]^(lst[[2]]+lst[[3]]*x)]]]]] /;
 Not[FalseQ[lst]]] /;
SimplifyFlag &&
Not[MatchQ[u,v_^n_. /; SumQ[v] && IntegerQ[n] && n>0]] &&
Not[MatchQ[u,v_^n_.*f_^(a_.+b_.*x) /; FreeQ[{a,b,f},x] && SumQ[v] && IntegerQ[n] && n>0]] &&
Not[MatchQ[u,1/(a_.+b_.*f_^(d_.+e_.*x)+c_.*f_^(g_.+h_.*x)) /;
    FreeQ[{a,b,c,d,e,f,g,h},x] && ZeroQ[g-2*d] && ZeroQ[h-2*e]]] &&
FalseQ[FunctionOfHyperbolic[u,x]] (* && u===ExpnExpand[u,x] *),

Int[u_,x_Symbol] :=
  Module[{lst=FunctionOfExponentialOfLinear[u,x]},
  Dist[1/(lst[[3]]*Log[lst[[4]]]),
       Subst[Int[Regularize[lst[[1]]/x,x],x],x,lst[[4]]^(lst[[2]]+lst[[3]]*x)]] /;
 Not[FalseQ[lst]]] /;
Not[MatchQ[u,v_^n_. /; SumQ[v] && IntegerQ[n] && n>0]] &&
Not[MatchQ[u,v_^n_.*f_^(a_.+b_.*x) /; FreeQ[{a,b,f},x] && SumQ[v] && IntegerQ[n] && n>0]] &&
Not[MatchQ[u,1/(a_.+b_.*f_^(d_.+e_.*x)+c_.*f_^(g_.+h_.*x)) /;
    FreeQ[{a,b,c,d,e,f,g,h},x] && ZeroQ[g-2*d] && ZeroQ[h-2*e]]] &&
FalseQ[FunctionOfHyperbolic[u,x]] (* && u===ExpnExpand[u,x] *) ]
```

Improper Binomial Subexpression Substitution Rules

- Derivation: Integration by substitution

```
Int[x_^m_.*f_^(a_.+b_.*x_^n_.),x_Symbol] :=
  -Subst[Int[f^(a+b*x^(-n))/x^(m+2),x],x,1/x] /;
FreeQ[{a,b,f},x] && IntegersQ[m,n] && n<0 && m<-1 && GCD[m+1,n]==1
```

- Derivation: Integration by substitution

```
Int[x_^m_.*f_[a_.+b_.*x_^n_]^p_.,x_Symbol] :=
  -Subst[Int[f[a+b*x^(-n)]^p/x^(m+2),x],x,1/x] /;
FreeQ[{a,b,f,p},x] && IntegersQ[m,n] && n<0 && m<-1 && GCD[m+1,n]==1
```

- Derivation: Algebraic simplification and distribution of fractional powers
- Basis: D[(a+b*x^n)^m/(x^(m*n)*(b+a/x^n)^m), x] == 0

```
Int[u_*(a_+b_.*x_^n_)^m_,x_Symbol] :=
  (a+b*x^n)^m/(x^(m*n)*(b+a/x^n)^m)*Int[u*x^(m*n)*(b+a/x^n)^m,x] /;
FreeQ[{a,b},x] && FractionQ[m] && IntegerQ[n] && n<-1 && u===ExpnExpand[u,x]
```

Fractional Power Subexpression Substitution Rules

- Derivation: Integration by substitution
- Basis: Int[f[(a+b*x)^(1/n), x], x] == n/b*Subst[Int[x^(n-1)*f[x, -a/b+x^n/b], x], x, (a+b*x)^(1/n)]

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{lst=SubstForFractionalPowerOfLinear[u,x]},
  ShowStep["","Int[f[(a+b*x)^(1/n),x],x]",
             "n/b*Subst[Int[x^(n-1)*f[x,-a/b+x^n/b],x],x,(a+b*x)^(1/n)]",Hold[
  Dist[lst[[2]]*lst[[4]],Subst[Int[lst[[1]],x],x,lst[[3]]^(1/lst[[2]])]]]] /;
 NotFalseQ[lst] && SubstForFractionalPowerQ[u,lst[[3]],x]] /;
SimplifyFlag,

Int[u_,x_Symbol] :=
  Module[{lst=SubstForFractionalPowerOfLinear[u,x]},
  Dist[lst[[2]]*lst[[4]],Subst[Int[lst[[1]],x],x,lst[[3]]^(1/lst[[2]])]] /;
 NotFalseQ[lst] && SubstForFractionalPowerQ[u,lst[[3]],x]]]
```

- Derivation: Integration by substitution
- Basis: Int[f[((a+b*x)/(c+d*x))^(1/n), x], x] ==
  n*(b*c-a*d)*Subst[Int[x^(n-1)*f[x, (-a+c*x^n)/(b-d*x^n)]/(b-d*x^n)^2, x], x, ((a+b*x)/(c+d*x))^(1/n)]

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{lst=SubstForFractionalPowerOfQuotientOfLinears[u,x]},
  ShowStep["","Int[f[((a+b*x)/(c+d*x))^(1/n),x],x]",
"n*(b*c-a*d)*Subst[Int[x^(n-1)*f[x,(-a+c*x^n)/(b-d*x^n)]/(b-d*x^n)^2,x],x,((a+b*x)/(c+d*x))^(1/n)]",
  Dist[lst[[2]]*lst[[4]],Subst[Int[lst[[1]],x],x,lst[[3]]^(1/lst[[2]])]]]] /;
 NotFalseQ[lst]] /;
SimplifyFlag,

Int[u_,x_Symbol] :=
  Module[{lst=SubstForFractionalPowerOfQuotientOfLinears[u,x]},
  Dist[lst[[2]]*lst[[4]],Subst[Int[lst[[1]],x],x,lst[[3]]^(1/lst[[2]])]] /;
 NotFalseQ[lst]]]
```

Linear Subexpression Substitution Rules

- Derivation: Integration by substitution
- Basis: Int[f[a+b*x], x] == Subst[Int[f[x], x], x, a+b*x]/b

```
Int[u_*(a_+b_.*x_)^m_.,x_Symbol] :=
  Dist[1/b,Subst[Int[x^m*Regularize[SubstFor[a+b*x,u,x],x],x],x,a+b*x]] /;
FreeQ[{a,b,m},x] && FunctionOfQ[a+b*x,u,x]
```

- Derivation: Integration by substitution
- Basis: Int[f[a+b*x, x], x] == Subst[Int[f[x, -a/b+x/b], x], x, a+b*x]/b

```
Int[x_^m_./(a_+b_.*(c_+d_.*x_)^n_), x_Symbol] :=
  Dist[1/d,Subst[Int[(-c/d+x/d)^m/(a+b*x^n),x],x,c+d*x]] /;
FreeQ[{a,b,c,d},x] && IntegersQ[m,n] && n>2
```

- Derivation: Integration by substitution

```
Int[(e_+f_.*x_)^m_.*(a_+b_.*(c_+d_.*x_)^n_)^p_, x_Symbol] :=
  Dist[(f/d)^m/d,Subst[Int[x^m*(a+b*x^n)^p,x],x,c+d*x]] /;
FreeQ[{a,b,c,d,e,f},x] && IntegersQ[m,n,p] && ZeroQ[d*e-c*f]
```

- Derivation: Integration by substitution
- Basis: Int[f[a+b*x, x], x] == Subst[Int[f[x, -a/b+x/b], x], x, a+b*x]/b

```
Int[(a_.+b_.*x_)^m_.*f_[c_.+d_.*x_]^p_.,x_Symbol] :=
  Dist[1/b,Subst[Int[x^m*f[c-a*d/b+d*x/b]^p,x],x,a+b*x]] /;
FreeQ[{a,b,c,d,m},x] && RationalQ[p] && Not[a===0 && b===1] &&
    MemberQ[{Sin,Cos,Sec,Csc,Sinh,Cosh,Sech,Csch},f]
```

- Derivation: Integration by substitution
- Basis: Int[f[a+b*x, x], x] == Subst[Int[f[x, -a/b+x/b], x], x, a+b*x]/b

```
Int[(a_.+b_.*x_)^m_*(c_.+d_.*x_+e_.*x_^2)^n_,x_Symbol] :=
  Dist[1/b,Subst[Int[x^m*(c-a*d/b+a^2*e/b^2+(d/b-2*a*e/b^2)*x+e*x^2/b^2)^n,x],x,a+b*x]] /;
FreeQ[{a,b,c,d,e,m,n},x] && FractionQ[n] && Not[a===0 && b===1]
```

Extended Integration by Parts Rules

- Derivation: Integration by parts
- Basis: Int[(g[x]+h[x])^n*g'[x],x] == (g[x]+h[x])^(n+1)/(n+1) - Int[(g[x]+h[x])^n*h'[x], x]

```
Int[(u_+x_^p_.)^n_*v_,x_Symbol] :=
  Module[{z=DerivativeDivides[u,v,x]},
  z*(u+x^p)^(n+1)/(n+1) -
  Dist[z*p,Int[x^(p-1)*(u+x^p)^n,x]] /;
 Not[FalseQ[z]]] /;
IntegerQ[p] && RationalQ[n] && NonzeroQ[n+1] && Not[AlgebraicFunctionQ[v,x]]
```

- Derivation: Integration by parts
- Basis: Int[f[x]*(g[x]+h[x])^n*g'[x],x] == f[x]*(g[x]+h[x])^(n+1)/(n+1) - Int[f[x]*(g[x]+h[x])^n*h'[x], x] - Int[f'[x]*(g[x]+h[x])^(n+1), x]/(n+1)

```
Int[x_^m_.*(u_+x_^p_.)^n_*v_,x_Symbol] :=
  Module[{z=DerivativeDivides[u,v,x]},
  z*x^m*(u+x^p)^(n+1)/(n+1) -
  Dist[z*p,Int[x^(m+p-1)*(u+x^p)^n,x]] -
  Dist[z*m/(n+1),Int[x^(m-1)*(u+x^p)^(n+1),x]] /;
 Not[FalseQ[z]]] /;
IntegersQ[m,p] && RationalQ[n] && NonzeroQ[n+1]
```

Logarithm Rules

- Reference: A&S 4.1.53
- Derivation: Integration by parts

```
Int[Log[u_],x_Symbol] :=
  x*Log[u] -
  Int[Regularize[x*D[u,x]/u,x],x] /;
InverseFunctionFreeQ[u,x]
```

- Reference: G&R 2.727.2
- Derivation: Integration by parts

```
Int[Log[u_]/x_,x_Symbol] :=
  Module[{v=D[u,x]/u},
  Log[u]*Log[x] -
  Int[Regularize[Log[x]*v,x],x] /;
 RationalFunctionQ[v,x]] /;
Not[BinomialTest[u,x] && BinomialTest[u,x][[3]]^2===1]
```

- Reference: G&R 2.727.2
- Derivation: Integration by parts

```
Int[Log[u_]/(a_+b_.*x_),x_Symbol] :=
  Module[{v=D[u,x]/u},
  Log[u]*Log[a+b*x]/b -
  Dist[1/b,Int[Regularize[Log[a+b*x]*v,x],x]] /;
 RationalFunctionQ[v,x]] /;
FreeQ[{a,b},x]
```

- Reference: G&R 2.725.1, A&S 4.1.54
- Derivation: Integration by parts

```
Int[(a_.+b_.*x_)^m_.*Log[u_],x_Symbol] :=
  Module[{v=D[u,x]/u},
  (a+b*x)^(m+1)*Log[u]/(b*(m+1)) -
  Dist[1/(b*(m+1)),Int[Regularize[(a+b*x)^(m+1)*v,x],x]] /;
FreeQ[{a,b,m},x] && NonzeroQ[m+1] && InverseFunctionFreeQ[u,x] &&
    Not[FunctionOfQ[x^(m+1),u,x]] &&
    FalseQ[PowerVariableExpn[u,m+1,x]]
```

- Derivation: Integration by parts

```
Int[v_*Log[u_],x_Symbol] :=
  Module[{w=Block[{ShowSteps=False,StepCounter=Null}, Int[v,x]]},
  w*Log[u] -
    Int[Regularize[w*D[u,x]/u,x],x] /;
 InverseFunctionFreeQ[w,x]] /;
InverseFunctionFreeQ[u,x] &&
    Not[MatchQ[v, x^m_. /; FreeQ[m,x]]] &&
    FalseQ[FunctionOfLinear[v*Log[u],x]]
```

Reciprocals of Quadratic Trinomials Expansion Rules

- Derivation: Algebraic expansion
- Basis: If q=Sqrt[-a/b], z/(a+b*z^2) == q/(2*(a+b*q*z)) - q/(2*(a-b*q*z))

```
Int[u_.*x_/(a_+b_.*x_^2),x_Symbol] :=
  Module[{q=Rt[-a/b,2]},
  Dist[q/2,Int[u/(a+b*q*x),x]] -
  Dist[q/2,Int[u/(a-b*q*x),x]]] /;
FreeQ[{a,b},x] && Not[MatchQ[u,r_*s_. /; SumQ[r]]] && Not[RationalFunctionQ[u,x]]
```

- Derivation: Algebraic expansion
- Basis: If q=Sqrt[b^2-4*a*c], z/(a+b*z+c*z^2) == (1+b/q)/(b+q+2*c*z) + (1-b/q)/(b-q+2*c*z))

```
Int[u_.*v_^m_./(a_+b_.*v_+c_.*w_),x_Symbol] :=
  Module[{q=Rt[b^2-4*a*c,2]},
  Dist[(1+b/q),Int[u*v^(m-1)/(b+q+2*c*v),x]] + Dist[(1-b/q),Int[u*v^(m-1)/(b-q+2*c*v),x]] /;
 NonzeroQ[q]] /;
FreeQ[{a,b,c},x] && RationalQ[m] && m==1 && ZeroQ[w-v^2] &&
Not[MatchQ[u,r_*s_. /; SumQ[r]]] && (Not[RationalFunctionQ[u,x]] || Not[RationalFunctionQ[v,x]])
```

- Derivation: Algebraic expansion
- Basis: If q=Sqrt[b^2-4*a*c], (d+e*z)/(a+b*z+c*z^2) == (e-2*c*d/q+b*e/q)/(b+q+2*c*z)) + (e+2*c*d/q-b*e/q)/(b-q+2*c*z)

```
Int[(d_.+e_.*v_)/(a_+b_.*v_+c_.*w_),x_Symbol] :=
  Module[{q=Rt[b^2-4*a*c,2]},
    Dist[e+(b*e-2*c*d)/q,Int[1/(b+q+2*c*v),x]] + Dist[e-(b*e-2*c*d)/q,Int[1/(b-q+2*c*v),x]] /;
 NonzeroQ[q]] /;
FreeQ[{a,b,c,d,e},x] && ZeroQ[w-v^2] && NonzeroQ[2*c*d-b*e] && Not[RationalFunctionQ[v,x]]
```

- Reference: G&R 2.161.1 a'
- Derivation: Algebraic expansion
- Basis: If q=Sqrt[b^2-4*a*c], 1/(a+b*z+c*z^2) == 2*c/(q*(b-q+2*c*z)) - 2*c/(q*(b+q+2*c*z))

```
Int[u_./(a_+b_.*v_+c_.*w_),x_Symbol] :=
  Module[{q=Rt[b^2-4*a*c,2]},
  Dist[2*c/q,Int[u/(b-q+2*c*v),x]] - Dist[2*c/q,Int[u/(b+q+2*c*v),x]] /;
 NonzeroQ[q]] /;
FreeQ[{a,b,c},x] && ZeroQ[w-v^2] && Not[MatchQ[u,v^m_ /; RationalQ[m]]] &&
Not[MatchQ[u,r_*s_. /; SumQ[r]]] && (Not[RationalFunctionQ[u,x]] || Not[RationalFunctionQ[v,x]])
```

General Algebraic Simplification Rules

- Derivation: Algebraic simplification

```
Int[u_,x_Symbol] :=
  Module[{v=SimplifyExpression[u,x]},
  Int[v,x] /;
 v=!=u ]
```

Piecewise Constant Extraction Rules

- Derivation: Piecewise constant extraction

```
Int[u_.*(v_^m_.*w_^n_.*t_^q_.)^p_,x_Symbol] :=
  Int[u*v^(m*p)*w^(n*p)*t^(p*q),x] /;
FreeQ[p,x] && Not[PowerQ[v]] && Not[PowerQ[w]] && Not[PowerQ[t]] &&
    ZeroQ[Simplify[(v^m*w^n*t^q)^p-v^(m*p)*w^(n*p)*t^(p*q)]]
```

- Derivation: Piecewise constant extraction

```
Int[u_.*(v_^m_.*w_^n_.*t_^q_.)^p_,x_Symbol] :=
  Module[{r=Simplify[(v^m*w^n*t^q)^p/(v^(m*p)*w^(n*p)*t^(p*q))],lst},
  ( lst=SplitFreeFactors[v^(m*p)*w^(n*p)*t^(p*q),x];
  r*lst[[1]]*Int[Regularize[u*lst[[2]],x],x] ) /;
 NonzeroQ[r-1]] /;
FreeQ[p,x] && Not[PowerQ[v] || PowerQ[w] || PowerQ[t] || FreeQ[v,x] || FreeQ[w,x] || FreeQ[t,x]]
```

General Algebraic Expansion Rules

- Author: Martin 13 July 2010
- Derivation: Algebraic expansion
- Basis: If n>0 is an integer, a+b*z^n == b*Product[z - (-a/b)^(1/n)*(-1)^(2*k/n), {k, 1, n}]
- Basis: If n>0 is an integer, a+b*z^n == a*Product[1 - z/((-a/b)^(1/n)*(-1)^(2*k/n)), {k, 1, 4}]
- Basis: If m and n are integers and 0<=m<n let q=(-a/b)^(1/n), then
  z^m/(a+b*z^n) == q^(m+1)*Sum[(-1)^(2*k*(m+1)/n)/(q*(-1)^(2*k/n) - z), {k, 1, n}]/(a*n)

```
Int[u_*x_^m_./(a_+b_.*x_^n_),x_Symbol] :=
  Module[{r=Numerator[Rt[-a/b,n]], s=Denominator[Rt[-a/b,n]]},
  Dist[r^(m+1)/(a*n*s^m), Sum[Int[u*(-1)^(2*k*(m+1)/n)/(r*(-1)^(2*k/n)-s*x),x],{k,1,n}]]] /;
FreeQ[{a,b},x] && IntegersQ[m,n] && 0<m<n && Not[AlgebraicFunctionQ[u,x]]
```

- Derivation: Algebraic expansion
- Basis: If n>0 is an integer let q=(-a/b)^(1/n), then 1/(a+b*z^n) == q*Sum[(-1)^(2*k/n)/(q*(-1)^(2*k/n) - z), {k, 1, n}]/(a*n)

```
Int[u_/(a_+b_.*x_^n_),x_Symbol] :=
  Module[{r=Numerator[Rt[-a/b,n]], s=Denominator[Rt[-a/b,n]]},
  Dist[r/(a*n), Sum[Int[u*(-1)^(2*k/n)/(r*(-1)^(2*k/n)-s*x),x],{k,1,n}]]] /;
FreeQ[{a,b},x] && OddQ[n] && n>1 && Not[AlgebraicFunctionQ[u,x]]
```

- Derivation: Algebraic expansion
- Basis: If n>0 is an integer, a+b*z^n == b*Product[z - (-a/b)^(1/n)*(-1)^(2*k/n), {k, 1, n}]
- Basis: If n>0 is an integer, z^(n-1)/(a+b*z^n) == Sum[1/(z - (-a/b)^(1/n)*(-1)^(2*k/n)), {k, 1, n}]/(b*n)

```
Int[u_.*v_^m_/(a_+b_.*v_^n_),x_Symbol] :=
  Dist[1/(b*n),Sum[Int[Together[u/(v-Rt[-a/b,n]*(-1)^(2*k/n))],x],{k,1,n}]] /;
FreeQ[{a,b},x] && OddQ[n] && n>1 && ZeroQ[m-n+1] &&
  Not[AlgebraicFunctionQ[u,x] && AlgebraicFunctionQ[v,x]]
```

- Derivation: Algebraic expansion
- Basis: If n>0 is an integer, a+b*z^n == a*Product[1 - z/((-a/b)^(1/n)*(-1)^(2*k/n)), {k, 1, 4}]
- Basis: If n>0 is an integer, 1/(a+b*z^n) == Sum[1/(1 - z/((-a/b)^(1/n)*(-1)^(2*k/n))), {k, 1, n}]/(a*n)

```
Int[u_./(a_+b_.*v_^n_),x_Symbol] :=
  Dist[1/(a*n),Sum[Int[Together[u/(1-v/(Rt[-a/b,n]*(-1)^(2*k/n)))],x],{k,1,n}]] /;
FreeQ[{a,b},x] && OddQ[n] && n>1 && Not[AlgebraicFunctionQ[u,x] && AlgebraicFunctionQ[v,x]]
```

- Derivation: Algebraic expansion

```
Int[u_,x_Symbol] :=
  Module[{v=ExpnExpand[u,x]},
  Int[v,x] /;
 v=!=u ]
```

Function of Linear Binomial Substitution Rules

- Derivation: Integration by substitution
- Basis: Int[f[1/(a+b*x)], x] == -Subst[Int[f[x]/x^2, x], x, 1/(a+b*x)]/b
- Basis: Int[f[(a+b*x)/(c+d*x)], x] == -Subst[Int[f[b/d+(a*d-b*c)/d*x]/x^2, x], x, 1/(c+d*x)]/d

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{lst=SubstForInverseLinear[u,x]},
  ShowStep["","Int[f[1/(a+b*x)],x]","-Subst[Int[f[x]/x^2,x],x,1/(a+b*x)]/b",Hold[
  -Dist[1/lst[[3]],Subst[Int[lst[[1]]/x^2,x],x,1/lst[[2]]]]]] /;
 NotFalseQ[lst]] /;
SimplifyFlag,

Int[u_,x_Symbol] :=
  Module[{lst=SubstForInverseLinear[u,x]},
  -Dist[1/lst[[3]],Subst[Int[lst[[1]]/x^2,x],x,1/lst[[2]]]] /;
 NotFalseQ[lst]]]
```

- Derivation: Integration by substitution
- Basis: Int[f[a+b*x], x] == Subst[Int[f[x], x], x, a+b*x]/b

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{lst=FunctionOfLinear[u,x]},
  ShowStep["","Int[f[a+b*x],x]","Subst[Int[f[x],x],x,a+b*x]/b",Hold[
  Dist[1/lst[[3]],Subst[Int[lst[[1]],x],x,lst[[2]]+lst[[3]]*x]]]] /;
 Not[FalseQ[lst]]] /;
SimplifyFlag,

Int[u_,x_Symbol] :=
  Module[{lst=FunctionOfLinear[u,x]},
  Dist[1/lst[[3]],Subst[Int[lst[[1]],x],x,lst[[2]]+lst[[3]]*x]] /;
 Not[FalseQ[lst]]]]
```

Negative Powers of Binomials Expansion Rules

- Derivation: Algebraic expansion
- Basis: If n>0 is even, 1/(a+b*z^n) == 2/(a*n)*Sum[1/(1-z^2/((-a/b)^(2/n)*(-1)^(4*k/n))), {k, 1, n/2}]

```
Int[u_./(a_+b_.*v_^n_),x_Symbol] :=
  Dist[2/(a*n),Sum[Int[Together[u/(1-v^2/(Rt[-a/b,n/2]*(-1)^(4*k/n)))],x],{k,1,n/2}]] /;
FreeQ[{a,b},x] && EvenQ[n] && n>2
```

- Derivation: Algebraic expansion
- Basis: If n>0 is even, a+b*z^n == a*Product[1-(-1)^(4*k/n)*(-b/a)^(2/n)*z^2, {k, 1, n/2}]

```
Int[u_.*(a_+b_.*v_^n_)^m_,x_Symbol] :=
  Dist[a^m,Int[u*Product[(1-(-1)^(4*k/n)*Rt[-b/a,n/2]*v^2)^m,{k,1,n/2}],x]] /;
FreeQ[{a,b},x] && IntegerQ[m] && m<-1 && EvenQ[n] && n>2 (* && NegQ[b/a] *)
```

- Derivation: Algebraic expansion
- Basis: If n>0 is an integer, a+b*z^n == b*Product[-(-a/b)^(1/n)*(-1)^(2*k/n) + z, {k, 1, n}]
- Basis: If n>0 is an integer, a+b*z^n == a*Product[1-(-1)^(2*k/n)*(-b/a)^(1/n)*z, {k, 1, n}]
- Basis: If n>0 is odd, a+b*z^n == a*Product[1+(-1)^(2*k/n)*(b/a)^(1/n)*z, {k, 1, n}]

```
Int[u_.*(a_+b_.*v_^n_)^m_,x_Symbol] :=
  Dist[a^m,Int[u*Product[(1+(-1)^(2*k/n)*Rt[b/a,n]*v)^m,{k,1,n}],x]] /;
FreeQ[{a,b},x] && IntegerQ[m] && m<-1 && OddQ[n] && n>1
```

Negative Powers of Trinomials Expansion Rules

- Derivation: Algebraic expansion
- Basis: a+b*z+c*z^2 == (b-Sqrt[b^2-4*a*c]+2*c*z)*(b+Sqrt[b^2-4*a*c]+2*c*z)/(4*c)

```
Int[u_.*(a_+b_.*v_+c_.*w_)^m_,x_Symbol] :=
  Dist[1/(4*c)^m,Int[u*(b-Sqrt[b^2-4*a*c]+2*c*v)^m*(b+Sqrt[b^2-4*a*c]+2*c*v)^m,x]] /;
FreeQ[{a,b,c},x] && IntegerQ[m] && m<0 && ZeroQ[w-v^2]
```

Integrand Normalization Rules

- Derivation: Algebraic simplification
- Note: Replace this rule with specific rules for each normalization.

```
Int[u_,x_Symbol] :=
  Module[{v=NormalForm[u,x]},
  Int[v,x] /;
 Not[v===u]]
```

Piecewise Constant Extraction Rules

- Derivation: Piecewise constant extraction
- Basis: D[(a*f[x]^m)^p/f[x]^(m*p), x] == 0

```
Int[u_.*(a_.*v_^m_.)^p_, x_Symbol] :=
  Module[{q=FractionalPart[p]},
  q=a^(p-q)*(a*v^m)^q/v^(m*q);
  If[FreeQ[Simplify[q],x],
    Simplify[q]*Int[u*v^(m*p),x],
  q*Int[u*v^(m*p),x]]] /;
FreeQ[{a,m},x] && FractionQ[p] && Not[ZeroQ[a-1] && ZeroQ[m-1]]
```

- Derivation: Piecewise constant extraction
- Basis: D[(f[x]^m)^p/f[x]^(m*p), x] == 0

```
Int[u_.*(v_^m_)^p_,x_Symbol] :=
  Simplify[(v^m)^p/v^(m*p)]*Int[Regularize[u*v^(m*p),x],x] /;
FreeQ[p,x] && Not[PowerQ[v]]
```

- Derivation: Piecewise constant extraction
- Basis: D[(a*f[x]^m*g[x]^n)^p/(f[x]^(m*p)*g[x]^(n*p)), x] == 0

```
Int[u_.*(a_.*v_^m_.*w_^n_.)^p_, x_Symbol] :=
  Module[{q=FractionalPart[p]},
  q=a^(p-q)*(a*v^m*w^n)^q/(v^(m*q)*w^(n*q));
  If[FreeQ[Simplify[q],x],
    Simplify[q]*Int[u*v^(m*p)*w^(n*p),x],
  q*Int[u*v^(m*p)*w^(n*p),x]]] /;
FreeQ[a,x] && RationalQ[{m,n,p}]
```

- Derivation: Piecewise constant extraction
- Basis: D[(f[x]^m*g[x]^n)^p/(f[x]^(m*p)*g[x]^(n*p)), x] == 0

```
Int[u_.*(v_^m_.*w_^n_.)^p_,x_Symbol] :=
  Module[{r=Simplify[(v^m*w^n)^p/(v^(m*p)*w^(n*p))],lst},
  If[ZeroQ[r-1],
    Int[u*v^(m*p)*w^(n*p),x],
  lst=SplitFreeFactors[v^(m*p)*w^(n*p),x];
  r*lst[[1]]*Int[Regularize[u*lst[[2]],x],x]]] /;
FreeQ[p,x] && Not[PowerQ[v]] && Not[PowerQ[w]]
```

Products of Fractional Powers Collection Rules

- Derivation: Collection of fractional powers
- Basis: D[f[x]^m/g[x]^m/(f[x]/g[x])^m, x] == 0
- Basis: Int[v^m/w^m, x] == v^m/w^m/(v/w)^m*Int[(v/w)^m, x]

```
Int[u_.*v_^m_*w_^n_,x_Symbol] :=
  Module[{q=Cancel[v/w]},
  (v^m*w^n)/q^m*Int[u*q^m,x] /;
 PolynomialQ[q,x]] /;
FractionQ[{m,n}] && m+n==0 && PolynomialQ[v,x] && PolynomialQ[w,x]
```

Fractional Power of Linear Subexpression Substitution Rules

- Derivation: Integration by substitution
- Basis: Int[f[(a+b*x)^(1/n), x], x] == n/b*Subst[Int[x^(n-1)*f[x, -a/b+x^n/b], x], x, (a+b*x)^(1/n)]

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{lst=SubstForFractionalPowerOfLinear[u,x]},
  ShowStep["","Int[f[(a+b*x)^(1/n),x],x]",
             "n/b*Subst[Int[x^(n-1)*f[x,-a/b+x^n/b],x],x,(a+b*x)^(1/n)]",Hold[
  Dist[lst[[2]]*lst[[4]],Subst[Int[lst[[1]],x],x,lst[[3]]^(1/lst[[2]])]]]] /;
 NotFalseQ[lst] (* && AlgebraicFunctionQ[lst[[1]],x] *) ] /;
SimplifyFlag,

Int[u_,x_Symbol] :=
  Module[{lst=SubstForFractionalPowerOfLinear[u,x]},
  Dist[lst[[2]]*lst[[4]],Subst[Int[lst[[1]],x],x,lst[[3]]^(1/lst[[2]])]] /;
 NotFalseQ[lst] (* && AlgebraicFunctionQ[lst[[1]],x] *) ]]
```

Quadratic Binomial Expansion Rules

- Derivation: Algebraic expansion
- Basis: 1/(a+b*z^2) == 1/(2*(a+b*Sqrt[-a/b]*z)) + 1/(2*(a-b*Sqrt[-a/b]*z))
- Note: This rule necessary because ExpnExpand cannot expand Sqrt[x + 1]/((1 - I*x)*(1 + I*x)).

```
Int[u_./(a_+b_.*v_^2),x_Symbol] :=
  Dist[1/2,Int[u/(a+b*Rt[-a/b,2]*v),x]] + Dist[1/2,Int[u/(a-b*Rt[-a/b,2]*v),x]] /;
FreeQ[{a,b},x] (* && Not[PositiveQ[-a/b]] *)
```

- Derivation: Algebraic expansion
- Basis: a+b*z^2 == a*(1+Sqrt[-b/a]*z)*(1-Sqrt[-b/a]*z)

```
Int[u_.*(a_+b_.*v_^2)^m_,x_Symbol] :=
  Dist[a^m,Int[u*(1+Rt[-b/a,2]*v)^m*(1-Rt[-b/a,2]*v)^m,x]] /;
FreeQ[{a,b},x] && IntegerQ[m] && (m<-1 || m==-1 && PositiveQ[-b/a])
```

Exponential Function Expansion Rules

- Derivation: Algebraic expansion
- Basis: f^(z+w) == f^z*f^w

```
Int[u_.*f_^(a_+v_)*g_^(b_+w_),x_Symbol] :=
  Dist[f^a*g^b,Int[u*f^v*g^w,x]] /;
FreeQ[{a,b,f,g},x] && Not[MatchQ[v,c_+t_ /; FreeQ[c,x]]] && Not[MatchQ[w,c_+t_ /; FreeQ[c,x]]]
```

- Derivation: Algebraic expansion
- Basis: f^(z+w) == f^z*f^w

```
Int[u_.*f_^(a_+v_),x_Symbol] :=
  Dist[f^a,Int[u*f^v,x]] /;
FreeQ[{a,f},x] && Not[MatchQ[v,b_+w_ /; FreeQ[b,x]]]
```

Trig Function Piecewise Constant Extraction Rules

- **Derivation: Piecewise constant extraction and algebraic expansion**

- **Basis: If** $a^2 - b^2 = 0$**, then** $\partial_z \frac{\sqrt{a+b\,\text{Sin}[z]}}{\text{Cos}\left[\frac{z}{2}\right]+\frac{a}{b}\,\text{Sin}\left[\frac{z}{2}\right]} = 0$

- **Rule: If** $a^2 - b^2 = 0 \bigwedge n - \frac{1}{2} \in \mathbb{Z}$**, then**

$$\int u\,(a+b\,\text{Sin}[c+d\,x])^n\,dx \longrightarrow \frac{\sqrt{a+b\,\text{Sin}[c+d\,x]}}{\text{Cos}\left[\frac{c}{2}+\frac{d\,x}{2}\right]+\frac{a}{b}\,\text{Sin}\left[\frac{c}{2}+\frac{d\,x}{2}\right]}\cdot$$
$$\left(\int u\,\text{Cos}\left[\frac{c}{2}+\frac{d\,x}{2}\right]\,(a+b\,\text{Sin}[c+d\,x])^{n-\frac{1}{2}}\,dx+\frac{a}{b}\int u\,\text{Sin}\left[\frac{c}{2}+\frac{d\,x}{2}\right]\,(a+b\,\text{Sin}[c+d\,x])^{n-\frac{1}{2}}\,dx\right)$$

- **Program code:**

```
Int[u_*(a_+b_.*Sin[c_.+d_.*x_])^n_,x_Symbol] :=
  Sqrt[a+b*Sin[c+d*x]]/(Cos[c/2+d*x/2]+a/b*Sin[c/2+d*x/2])*
    (Int[u*Cos[c/2+d*x/2]*(a+b*Sin[c+d*x])^(n-1/2),x] +
     a/b*Int[u*Sin[c/2+d*x/2]*(a+b*Sin[c+d*x])^(n-1/2),x]) /;
FreeQ[{a,b,c,d},x] && ZeroQ[a^2-b^2] && IntegerQ[n-1/2]
```

- **Derivation: Piecewise constant extraction**

- **Basis:** $\partial_z \frac{\sqrt{a+a\,\text{Cos}[z]}}{\text{Cos}\left[\frac{z}{2}\right]} = 0$

- **Rule: If** $n - \frac{1}{2} \in \mathbb{Z}$**, then**

$$\int u\,(a+a\,\text{Cos}[c+d\,x])^n\,dx \longrightarrow \frac{\sqrt{a+a\,\text{Cos}[c+d\,x]}}{\text{Cos}\left[\frac{c}{2}+\frac{d\,x}{2}\right]}\int u\,\text{Cos}\left[\frac{c}{2}+\frac{d\,x}{2}\right]\,(a+a\,\text{Cos}[c+d\,x])^{n-\frac{1}{2}}\,dx$$

- **Program code:**

```
Int[u_*(a_+b_.*Cos[c_.+d_.*x_])^n_,x_Symbol] :=
  Sqrt[a+b*Cos[c+d*x]]/Cos[c/2+d*x/2]*Int[u*Cos[c/2+d*x/2]*(a+b*Cos[c+d*x])^(n-1/2),x] /;
FreeQ[{a,b,c,d},x] && ZeroQ[a-b] && IntegerQ[n-1/2]
```

- **Derivation: Piecewise constant extraction**

- **Basis:** $\partial_z \frac{\sqrt{a-a\,\text{Cos}[z]}}{\text{Sin}\left[\frac{z}{2}\right]} = 0$

- **Rule: If** $n-\frac{1}{2} \in \mathbb{Z}$, **then**

$$\int u\,(a-a\,\text{Cos}[c+d\,x])^n\,dx \longrightarrow \frac{\sqrt{a-a\,\text{Cos}[c+d\,x]}}{\text{Sin}\left[\frac{c}{2}+\frac{d\,x}{2}\right]}\int u\,\text{Sin}\left[\frac{c}{2}+\frac{d\,x}{2}\right]\,(a-a\,\text{Cos}[c+d\,x])^{n-\frac{1}{2}}\,dx$$

- **Program code:**

```
Int[u_*(a_+b_.*Cos[c_.+d_.*x_])^n_,x_Symbol] :=
  Sqrt[a+b*Cos[c+d*x]]/Sin[c/2+d*x/2]*Int[u*Sin[c/2+d*x/2]*(a+b*Cos[c+d*x])^(n-1/2),x] /;
FreeQ[{a,b,c,d},x] && ZeroQ[a+b] && IntegerQ[n-1/2]
```

- **Derivation: Piecewise constant extraction and algebraic expansion**

- **Basis: If** $a^2-b^2-c^2=0$, **then** $\partial_z \frac{\sqrt{a+b\,\text{Cos}[z]+c\,\text{Sin}[z]}}{c\,\text{Cos}\left[\frac{z}{2}\right]+(a-b)\,\text{Sin}\left[\frac{z}{2}\right]} = 0$

- **Rule: If** $a^2-b^2-c^2=0 \bigwedge n-\frac{1}{2} \in \mathbb{Z}$, **then**

$$\int u\,(a+b\,\text{Cos}[d+e\,x]+c\,\text{Sin}[d+e\,x])^n\,dx \longrightarrow$$

$$\frac{c\sqrt{a+b\,\text{Cos}[d+e\,x]+c\,\text{Sin}[d+e\,x]}}{c\,\text{Cos}\left[\frac{d}{2}+\frac{e\,x}{2}\right]+(a-b)\,\text{Sin}\left[\frac{d}{2}+\frac{e\,x}{2}\right]}\int u\,\text{Cos}\left[\frac{d}{2}+\frac{e\,x}{2}\right]\,(a+b\,\text{Cos}[d+e\,x]+c\,\text{Sin}[d+e\,x])^{n-\frac{1}{2}}\,dx +$$

$$\frac{(a-b)\sqrt{a+b\,\text{Cos}[d+e\,x]+c\,\text{Sin}[d+e\,x]}}{c\,\text{Cos}\left[\frac{d}{2}+\frac{e\,x}{2}\right]+(a-b)\,\text{Sin}\left[\frac{d}{2}+\frac{e\,x}{2}\right]}\int u\,\text{Sin}\left[\frac{d}{2}+\frac{e\,x}{2}\right]\,(a+b\,\text{Cos}[d+e\,x]+c\,\text{Sin}[d+e\,x])^{n-\frac{1}{2}}\,dx$$

- **Program code:**

```
Int[u_*(a_+b_.*Cos[d_.+e_.*x_]+c_.*Sin[d_.+e_.*x_])^n_,x_Symbol] :=
  Sqrt[a+b*Cos[d+e*x]+c*Sin[d+e*x]]/(c*Cos[d/2+e*x/2]+(a-b)*Sin[d/2+e*x/2])*
    Dist[c,Int[u*Cos[d/2+e*x/2]*(a+b*Cos[d+e*x]+c*Sin[d+e*x])^(n-1/2),x]] +
  Sqrt[a+b*Cos[d+e*x]+c*Sin[d+e*x]]/(c*Cos[d/2+e*x/2]+(a-b)*Sin[d/2+e*x/2])*
    Dist[a-b,Int[u*Sin[d/2+e*x/2]*(a+b*Cos[d+e*x]+c*Sin[d+e*x])^(n-1/2),x]] /;
FreeQ[{a,b,c,d,e},x] && ZeroQ[a^2-b^2-c^2] && IntegerQ[n-1/2]
```

Tangent $\theta/2$ Trig Substitution Rules

- Reference: CRC 484
- Derivation: Integration by substitution
- Basis: Sin[x] == 2*Tan[x/2]/(1+Tan[x/2]^2)
- Basis: Cos[x] == (1-Tan[x/2]^2)/(1+Tan[x/2]^2)
- Basis: 1+Tan[x/2]^2 == Tan'[x/2]

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  ShowStep["","Int[f[Sin[x],Cos[x]],x]",
           "2*Subst[Int[f[2*x/(1+x^2),(1-x^2)/(1+x^2)]/(1+x^2),x],x,Tan[x/2]]",Hold[
  Dist[2,
   Subst[Int[Regularize[SubstForTrig[u,2*x/(1+x^2),(1-x^2)/(1+x^2),x,x]/(1+x^2),x],x],x,Tan[x/2]]]]]
SimplifyFlag && FunctionOfTrigQ[u,x,x],

Int[u_,x_Symbol] :=
  Dist[2,
   Subst[Int[Regularize[SubstForTrig[u,2*x/(1+x^2),(1-x^2)/(1+x^2),x,x]/(1+x^2),x],x],x,Tan[x/2]]] /
FunctionOfTrigQ[u,x,x]]
```

Hyperbolic Tangent θ/2 Substitution Rules

- Derivation: Integration by substitution
- Basis: Sinh[x] == 2*Tanh[x/2]/(1-Tanh[x/2]^2)
- Basis: Cosh[x] == (1+Tanh[x/2]^2)/(1-Tanh[x/2]^2)
- Basis: 1-Tanh[x/2]^2 == Tanh'[x/2]

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  ShowStep["","Int[f[Sinh[x],Cosh[x]],x]",
            "2*Subst[Int[f[2*x/(1-x^2),(1+x^2)/(1-x^2)]/(1-x^2),x],x,Tanh[x/2]]",Hold[
  Dist[2,
   Subst[Int[Regularize[SubstForHyperbolic[u,2*x/(1-x^2),(1+x^2)/(1-x^2),x,x]/(1-x^2),x],x],x,Tanh[x,
SimplifyFlag && FunctionOfHyperbolicQ[u,x,x],

Int[u_,x_Symbol] :=
  Dist[2,
   Subst[Int[Regularize[SubstForHyperbolic[u,2*x/(1-x^2),(1+x^2)/(1-x^2),x,x]/(1-x^2),x],x],x,Tanh[x,
FunctionOfHyperbolicQ[u,x,x]]
```

Euler's Quadratic Subexpresion Substitution Rules

- Reference: G&R 2.251.1
- Derivation: Integration by Euler substitution for a>0

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{lst=FunctionOfSquareRootOfQuadratic[u,x]},
  ShowStep["","Int[f[Sqrt[a+b*x+c*x^2],x],x]",
            "2*Subst[Int[f[(c*Sqrt[a]-b*x+Sqrt[a]*x^2)/(c-x^2),(-b+2*Sqrt[a]*x)/(c-x^2)]*
                (c*Sqrt[a]-b*x+Sqrt[a]*x^2)/(c-x^2)^2,x],x,(-Sqrt[a]+Sqrt[a+b*x+c*x^2])/x]",
            Hold[Dist[2,Subst[Int[lst[[1]],x],x,lst[[2]]]]]] /;
 Not[FalseQ[lst]] && lst[[3]]===1] /;
SimplifyFlag,

Int[u_,x_Symbol] :=
  Module[{lst=FunctionOfSquareRootOfQuadratic[u,x]},
  Dist[2,Subst[Int[lst[[1]],x],x,lst[[2]]]] /;
 Not[FalseQ[lst]]]]
```

- Reference: G&R 2.251.2
- Derivation: Integration by Euler substitution for c>0

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{lst=FunctionOfSquareRootOfQuadratic[u,x]},
  ShowStep["","Int[f[Sqrt[a+b*x+c*x^2],x],x]",
            "2*Subst[Int[f[(a*Sqrt[c]+b*x+Sqrt[c]*x^2)/(b+2*Sqrt[c]*x),(-a+x^2)/(b+2*Sqrt[c]*x)]*
                (a*Sqrt[c]+b*x+Sqrt[c]*x^2)/(b+2*Sqrt[c]*x)^2,x],x,Sqrt[c]*x+Sqrt[a+b*x+c*x^2]]",
            Hold[Dist[2,Subst[Int[lst[[1]],x],x,lst[[2]]]]]] /;
 Not[FalseQ[lst]] && lst[[3]]===2] /;
SimplifyFlag,

Int[u_,x_Symbol] :=
  Module[{lst=FunctionOfSquareRootOfQuadratic[u,x]},
  Dist[2,Subst[Int[lst[[1]],x],x,lst[[2]]]] /;
 Not[FalseQ[lst]]]]
```

- Reference: G&R 2.251.3
- Derivation: Integration by Euler substitution

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{lst=FunctionOfSquareRootOfQuadratic[u,x]},
  ShowStep["","Int[f[Sqrt[a+b*x+c*x^2],x],x]",
           "-2*Sqrt[b^2-4*a*c]*Subst[Int[f[-Sqrt[b^2-4*a*c]*x/(c-x^2),
              (b*c+c*Sqrt[b^2-4*a*c]+(-b+Sqrt[b^2-4*a*c])*x^2)/(-2*c*(c-x^2))]*x/(c-x^2)^2,x],
               x,2*c*Sqrt[a+b*x+c*x^2]/(b-Sqrt[b^2-4*a*c]+2*c*x)]",
           Hold[Dist[2,Subst[Int[lst[[1]],x],x,lst[[2]]]]]] /;
 Not[FalseQ[lst]] && lst[[3]]===3] /;
SimplifyFlag,

Int[u_,x_Symbol] :=
  Module[{lst=FunctionOfSquareRootOfQuadratic[u,x]},
  Dist[2,Subst[Int[lst[[1]],x],x,lst[[2]]]] /;
 Not[FalseQ[lst]]]]
```

Inverse Function Substitution Rules

- Derivation: Integration by substitution
- Basis: f[z]/Sqrt[1-z^2] == f[Sin[ArcSin[z]]]*ArcSin'[z]

```
Int[u_*(1-(a_.+b_.*x_)^2)^n_.,x_Symbol] :=
  Module[{tmp=InverseFunctionOfLinear[u,x]},
  Dist[1/b,Subst[Int[Regularize[SubstForInverseFunction[u,tmp,x]*Cos[x]^(2*n+1),x],x],x,tmp]] /;
 NotFalseQ[tmp] && tmp===ArcSin[a+b*x]] /;
FreeQ[{a,b},x] && IntegerQ[2*n]
```

- Derivation: Integration by substitution
- Basis: f[z]/Sqrt[1-z^2] == -f[Cos[ArcCos[z]]]*ArcCos'[z]

```
Int[u_*(1-(a_.+b_.*x_)^2)^n_.,x_Symbol] :=
  Module[{tmp=InverseFunctionOfLinear[u,x]},
  -Dist[1/b,Subst[Int[Regularize[SubstForInverseFunction[u,tmp,x]*Sin[x]^(2*n+1),x],x],x,tmp]] /;
 NotFalseQ[tmp] && tmp===ArcCos[a+b*x]] /;
FreeQ[{a,b},x] && IntegerQ[2*n]
```

- Derivation: Integration by substitution
- Basis: f[z]/Sqrt[1+z^2] == f[Sinh[ArcSinh[z]]]*ArcSinh'[z]

```
Int[u_*(1+(a_.+b_.*x_)^2)^n_.,x_Symbol] :=
  Module[{tmp=InverseFunctionOfLinear[u,x]},
  Dist[1/b,Subst[Int[Regularize[SubstForInverseFunction[u,tmp,x]*Cosh[x]^(2*n+1),x],x],x,tmp]] /;
 NotFalseQ[tmp] && tmp===ArcSinh[a+b*x]] /;
FreeQ[{a,b},x] && IntegerQ[2*n]
```

- Derivation: Integration by substitution
- Basis: If h[g[x]] == x, Int[f[x, g[a+b*x]], x] == Subst[Int[f[-a/b+h[x]/b, x]*h'[x], x], x, g[a+b*x]]/b

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{lst=SubstForInverseFunctionOfLinear[u,x]},
  ShowStep["If h[g[x]]==x","Int[f[x,g[a+b*x]],x]",
    "Subst[Int[f[-a/b+h[x]/b,x]*h'[x],x],x,g[a+b*x]]/b",Hold[
  Dist[1/lst[[3]],Subst[Int[lst[[1]],x],x,lst[[2]]]]]] /;
 NotFalseQ[lst]] /;
SimplifyFlag && Not[NotIntegrableQ[u,x]],

Int[u_,x_Symbol] :=
  Module[{lst=SubstForInverseFunctionOfLinear[u,x]},
  Dist[1/lst[[3]],Subst[Int[lst[[1]],x],x,lst[[2]]]] /;
 NotFalseQ[lst]] /;
Not[NotIntegrableQ[u,x]]]
```

- Derivation: Integration by substitution
- Basis: If h[g[x]] == x,
  Int[f[x, g[(a+b*x)/(c+d*x)]], x] == (b*c-a*d)*Subst[Int[f[(-a+c*h[x])/(b-d*h[x]), x]*h'[x]/(b-d*h[x])^2, x], x, g[(a+b*x)/(c+d*x)]]

```
If[ShowSteps,

Int[u_,x_Symbol] :=
  Module[{lst=SubstForInverseFunctionOfQuotientOfLinears [u,x]},
  ShowStep["If h[g[x]]==x","Int[f[x,g[(a+b*x)/(c+d*x)]],x]",
"(b*c-a*d)*Subst[Int[f[(-a+c*h[x])/(b-d*h[x]),x]*h'[x]/(b-d*h[x])^2,x],x,g[(a+b*x)/(c+d*x)]]",Hold[
  Dist[lst[[3]],Subst[Int[lst[[1]],x],x,lst[[2]]]]]] /;
 NotFalseQ[lst]] /;
SimplifyFlag && Not[NotIntegrableQ[u,x]],

Int[u_,x_Symbol] :=
  Module[{lst=SubstForInverseFunctionOfQuotientOfLinears [u,x]},
  Dist[lst[[3]],Subst[Int[lst[[1]],x],x,lst[[2]]]] /;
 NotFalseQ[lst]] /;
Not[NotIntegrableQ[u,x]]]
```